THE DEVELOPMENT OF A HARMONIC CONTENT AND QUALITY OF ELECTRICITY SUPPLY MEASURING SYSTEM INCORPORATING SCADA PROCESSING

FREDERIK ANTONIE GROBLER

Thesis submitted in fulfilment of the requirements for the

DOCTOR TECHNOLOGIAE: ENGINEERING: ELECTRICAL

In the

School of Electrical and Computer Systems Engineering

of the

Faculty of Engineering, Information and Communication Technology

At the

Central University of Technology,

Free State

Promoter: Prof. G. D. Jordaan. D.Tech. (Eng.)

Bloemfontein
November 2005

DECLARATION

I, **FREDERIK ANTONIE GROBLER**, identity number [redacted], and student number **8809378** do hereby declare that this research project which has been submitted to the Central University of Technology, Free State for the Degree **DOCTOR TECNOLOGIAE: ENGINEERING: ELECTRICAL,** is my own independent work; and complies with the Code of Academic Integrity, as well as other relevant policies, procedures, rules and regulations of the Central University of Technology, Free State; and has not been submitted before by any other person in fulfilment (or partial fulfilment) of the requirements for the attainment of any qualification.

_____ _____

SIGNATURE OF STUDENT DATE

ACKNOWLEDGEMENTS

I had a great deal of support during the course of this research. In this regard I would like to thank:

Prof. G.D. Jordaan, for his supervision, guidance as well as excellent advice and assistance.

André Nel, for his dedication and support during the design and experimental phase of this research.

Wimpie van der Merwe and the EDSS SCADA engineers, for their assistance and support when performing tests and configuration changes on the ENMAC system.

Eskom, for assisting with the funds to enable the successful completion of this research.

Finally, acknowledgement to the **members of the profession and industry of electrical engineering,** whether as technicians, technologists, engineers or lecturers; all for the profound researches and practical application in everyday life that made this research possible.

Today's mighty oak
is just
yesterdays nut
that held it's ground.

- David Icke -

SUMMARY

When Thomas Edison invented his carbon filament lamp in 1879, gas shares fell overnight. A committee of inquiry was set up to examine the future possibilities of the new method of lighting, and had reached the conclusion that electric light in the home was fanciful and absurd. Today electric light burns in practically every house in the civilised world, with many great advances in the production and use of electricity and electric power supplied by various utilities.

The objective of the electric utility to deliver pure sinusoidal voltage at fairly constant magnitude throughout their system is complicated by the fact that there are currently loads on the system that produce harmonic voltages, which result in distorted voltages and currents that can adversely impact on the system performance in different ways. Because the numbers of harmonic producing loads have increased over the years, it has become necessary to address their influence, when making any additions or changes to an installation.

Quality of supply measurements have long been used to characterise non-linearity on the power system, and have traditionally been measured with expensive portable analysers. A potentially faster, more integrated, and more flexible solution to measure the harmonics with a Supervisory System is accomplished by this research.

Any script which aspired to cover in full detail the whole field of a subject so enormous as techniques to measure the quality of electricity supply on a SCADA system, would hardly be practical in less than a few volumes. The pretensions of this research are both modest and of a more immediate value to the reader.

UITTREKSEL

Met die ontdekking van die eerste koolstofdraad lamp in 1879 deur Thomas Edison, het die aandele in gas oornag baie sterk gedaal. ‘n Raad van ondersoek is saamgestel om die toekomstige moontlikhede van hierdie nuwe metode van beligting te ondersoek. Dit het tot die gevolgtrekking gekom dat die gebruik van elektriese ligte in die huis absurd en uitspattig is. Tog, vandag brand elektriese lig in byna elke huis van die moderne wêreld, met vooruitgang in die produksie en die gebruik van elektrisiteit en elektriese krag wat deur verskeie verskaffers voorsien word.

Die oogmerk van die elektrisiteitsverskaffer is om ‘n suiwer, redelik konstante elektriese spanning te lewer. Dit word gekompliseer deur die feit dat daar laste op die stelsel is wat harmonieke versteurings opwek, met die gevolg dat vervormde spannings en strome die elektriese stelsel se betroubare verrigting op verskillende maniere kan beïnvloed. Soos wat die harmoniek-opwekkende laste vermeerder, het dit nodig geraak om die invloed daarvan te oorweeg wanneer enige veranderings of byvoegings tot die installasies gemaak word.

Verskillende tegnieke om die kwaliteit van elektriese toevoer te meet is al voorheen gebruik om nie-lineariteit van netwerke te bepaal. Dit is in die verlede gewoonlik met baie gesofistikeerde en duur instrumente gemeet. ‘n Vinniger en meer geïntegreerde oplossing om harmonieke en kwaliteit van kragtoevoer deur die gebruik van Toesigbeheer stelsels te monitor word deur hierdie navorsing aangespreek.

Enige proefskrif wat ‘n onderwerp van so ‘n enorme omvang breedvoerig aanspreek, veral tegnieke bevattende die meet van elektriese toevoerkwaliteit deur middel van Toesighoudende apparaat, sal onprakties wees tensy dit minstens ‘n paar volumes beslaan. Dit is egter nie die doel van hierdie navorsing nie, maar die oogmerk van hierdie navorsing is slegs om van beskeie en onmiddelike waarde tot die leser te wees.

CONTENTS

CHAPTER 1

INTRODUCTION

CHAPTER 2

REVIEW OF ELECTRICAL POWER SUPPLY DISTORTION AND DETECTION TECHNIQUES

CHAPTER 3

MEASUREMENT SPECIFICATIONS AND DATA PROTOCOL ANALYSIS

CHAPTER 4

METHODS & TECHNIQUES

CHAPTER 5

ENMAC HARDWARE AND SOFTWARE CONFIGURATIONS

CHAPTER 6

HARDWARE & SOFTWARE EVALUATION

CHAPTER 7

SUMMARY AND CONCLUSIONS

APPENDIXES

APPENDIX 1.

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

CHAPTER 2

REVIEW OF ELECTRICAL POWER SUPPLY DISTORTION AND DETECTION TECHNIQUES

CHAPTER 3

MEASUREMENT SPECIFICATIONS AND DATA PROTOCOL ANALYSIS

CHAPTER 4

METHODS AND TECHNIQUES

CHAPTER 5

ENMAC HARDWARE AND SOFTWARE CONFIGURATIONS

CHAPTER 6

HARDWARE & SOFTWARE EVALUATION

LIST OF TABLES

CHAPTER 3

MEASUREMENT SPECIFICATIONS AND DATA PROTOCOL ANALYSIS.

CHAPTER 4

METHODS AND TECHNIQUES

CHAPTER 5

ENMAC HARDWARE AND SOFTWARE CONFIGURATIONS.

CHAPTER 6

HARDWARE & SOFTWARE EVALUATION

LIST OF FORMULAE

CHAPTER 2

REVIEW OF ELECTRICAL POWER SUPPLY DISTORTION AND DETECTION TECHNIQUES

CHAPTER 3

MEASUREMENT SPECIFICATIONS AND DATA PROTOCOL ANALYSIS

CHAPTER 4

METHODS AND TECHNIQUES

CHAPTER 6

ABBREVIATIONS

ASCII – American Standard Code for Information Interchange.

CE – Calculation Engine.

DNP-03 – Distribution Network Protocol version 3.

DRTU – Distribution Remote Terminal Unit.

DSP – Digital Signal Processing.

ENMAC – Electricity Network Management and Control system.

ERTU – Enhanced Remote Terminal Unit.

FEP – Front End Processor.

FFSK – Fast Frequency Shift Keying.

FFT – Fast Fourier Transform.

GRTV – Graphical Real Time Value.

GUI – Graphical User Interface.

IED – Intelligent Electronic Device.

IIR - Infinite Impulse Response

LAN - Local Area Network.

MTIO – Multi Input / Output card.

NER – National Energy Regulator.

NMS – Network Management Server.

NRS – National Regulatory Standards.

OW – Operator Workstation.

PEHMI – Portable Engineering Human Machine Interface.

PUTU – Programmable Universal Telecontrol Unit.

QOS – Quality of Supply.

QUICC – Quad Integrated Communication Controller.

RDBMS – Relational Database Management System.

RTAP – Real Time Application.

RTDB – Real Time Database.

RTDN – Real Time Data Network.

RTU – Remote Terminal Unit.

SCADA – Supervisory Control and Data Acquisition.

SCAMNGR – SCADA Manager.

TCP – Transmission Control Protocol.

THCD – Total Harmonic Current Distortion.

THD – Total Harmonic Distortion.

THVD – Total Harmonic Voltage Distortion.

TMNGR – Transaction Manager.

TSCPA – Two Shot Control Pre-select Acknowledge.

UNICON – Universal Network Configurator.

VT – Video Terminal.

WAN – Wide Area Network.

CHAPTER 1

INTRODUCTION

Automated and remote control of servo devices in our physical environment is a step on the way to supremacy, and the argument is that technology is the answer to man's fundamental desires. Without any doubt, the use of contemporary advanced electronic equipment has transformed our daily lives, but regrettably, the same technology has also changed the load characteristics of most modern electrical energy facilities.

1.1. The fundamentals and significance of the quality of electricity supply

As more advanced computer and electronic equipment are added to the existing power network, new phenomena appear, involving distortion and pollution of the power system. This phenomenon is concealed in most power lines, and existing supervisory apparatus fails to recognise this. This occurrence is described as voltage harmonics, voltage dips and voltage surges. In response to this, various techniques are used to support Supervisory Control and Data Acquisition (SCADA) systems to be able to recognise this pollution. This research investigates the latter, and strives to produce a low cost, accurate and flexible energy measurement package, running on most operational SCADA systems.

The issue of power quality is at present more important to industry, commerce and the home consumer than it was a few decades ago. Because the number of electrical devices that act as non-linear loads has increased significantly in the past decade, the injection of

harmonic content into the electricity network has increased noticeably. With advances in power electronics, harmonic-producing equipment has become widespread in their use. The need to measure, understand and act on the effects of harmonic and other forms of electricity distortion is ever more apparent given today's consumer driven market, in the midst of the expanding use of electronic circuitry.

Power quality is simply a product of the interaction of electrical power with electrical equipment. If electrical equipment operates correctly and reliably without being damaged or stressed, we would say that the electrical power is of good quality. On the other hand, if the electrical equipment malfunctions, is unreliable, or is damaged during normal usage, we would suspect that the power quality is poor.

The effect that harmonics have on electrical equipment, in particular sensitive electronic equipment, has only become apparent in the last number of years. Therefore, there is a need to investigate and measure the amount of harmonic content in electrical systems, in order to find solutions to remedy the situation so that adverse effects and added costs are avoided.

Electronic equipment incorporating integrated circuits and microprocessors is widely used in numerous critical situations today, and is most vulnerable to power disturbances. These disturbances can be introduced via the transmission line, or generated internally within the consumer's own facility. Types of power line disturbances can range from under-voltage and over-voltage, up to unwanted generally unidentified frequency variations like harmonics. Voltage distortion has long been recognised as being able to

interfere with telecommunication and control systems, to increase losses in circuits and equipment, and to cause overloading of rotating plant and capacitors, the latter being particularly susceptible to damage [17, p. 6]. Non-stop disturbances other than long-lasting over-voltage or under-voltage primarily manifest themselves in the form of harmonic distortion [44, pp. 60- 61, pp. 263-264].

According to Dugan and McGranaghan [10, p. 6] and Enslin [13, pp. 29-37], the most common sources of harmonic distortion may include:

- Rectifiers.
- Variable speed motor drives.
- Arc furnaces & arc welders.
- Transformer excitation current.
- Uninterrupted power supplies.
- Personal computer power supplies.
- Fluorescent lamp ballast's, etc.

Electronic loads have earned the name "non-linear load" to describe the way they draw power, and this term is commonly used to describe the switch-mode power supply found in personal computers. These non-linear loads are the primary source of harmonic currents and voltages [13, p. 37].

Plants that have a large proportion of these harmonic-producing devices may be susceptible to harmonic distortion problems. The source of these disturbances can be

from within the plant itself, or from external sources, but, whatever the source of disturbances, the effects must be minimised or eliminated. Failure to do so will result in an increase in downtime, lost production, increased cost of equipment maintenance and losses due to incorrect Watt-hour measurements [10, p. 35].

Fig. 1.1. ABB Powertech harmonic filter bank.

Applying surge protectors can minimise the effect of transient disturbances, motor starting, load sequencing, etc, but disturbance reduction will not reduce the effects of continuous disturbances. Reduction of continuous disturbances due to harmonics often requires the use of harmonic filters as shown in figure 1.1, which can be very expensive. In most instances, power factor correction capacitors can be installed in the form of a harmonic filter bank to provide both power factor correction and harmonic filtering capabilities. There has been an increasing concern about harmonic distortion and its effects on the power system lately. Some of the adverse effects of concentrated non-linear loads upon a facility are:

- Overheated transformers due to large magnetic fields emanating from transformers [13, p. 56].
- Over-voltage problems and decreased distribution capacity [13, p. 57].
- Excessive neutral return currents and high levels of neutral-to-ground voltage [13, p. 58].
- Power-factor rate penalties and metering problems [13, p. 59].
- Nuisance tripping of circuit breakers [13, p. 60].
- Voltage distortion within the facility can take place [13, p. 61].

1.2. The aim of this investigation

Most power supply utilities are extremely concerned with the supply of high quality uninterrupted electricity supply at a moderate cost to their consumers. This goal is affected by the presence of all the different types of distortion in power systems.

Power line disturbances are usually random phenomena, which do not occur in any kind of a fixed pattern with time [13, p. 13]. Test monitoring is a cumbersome task, and should be carried out at least for a period of two weeks or long enough for every type of potential disturbance to be discovered. The most desirable situation is therefore to install continuous monitoring equipment, which allows immediate response to be taken in the event of excessive pollution or bad power-quality areas, and incorporate it with a SCADA system. The development of a system for measuring such disturbances was the predominant objective of this research.

1.3. Hypothesis

It is now possible to:

1) Measure distortion and pollution levels on the power network via Supervisory Control and Data Acquisition (SCADA) systems.
2) To locate the sources of this contamination with the aid of the SCADA system.
3) To use the SCADA system to assist in eliminating sources of power pollution on the electrical power network.
4) Historical information of these contaminated areas and equipment can be kept on the SCADA database for future references and planning purposes.

1.4. The significance of this research

The implication of this research project significantly depends on the eventual, extensive use of the results. If the results and the designs proposed during this research are implemented on significant SCADA systems used in the SCADA environment in South Africa, the following can be obtained from the Master station:

- *Abnormal alarm alert:* Some of the premature failures of power system plant are actually the result of damage due to inadequate monitoring and metering. With the implementation of the Harmonic content and electricity quality of power supply measuring system on a SCADA system, alarms can be expected

to put the control engineers on the alert to abnormal conditions, for immediate action.

- *Power line disturbances can now be located*: Utilising the results of this research project will not only enable the SCADA system to be available to monitor power line disturbances and power quality, but will also assist in locating possible sources of harmonics. This can be done by using the SCADA system, (to isolate problem areas using alternative feeder sources and ring feed capabilities) in combination with the Quality of Supply (QOS) measuring device designed for this project.
- *Easy access to information:* By making use of the results obtained by this research, operations staff can see one-line or pictorial views of the power parameters. Engineering staff can also view advanced power features; power quality sequence of events with minimum and maximum time-tagged events. Customers can be informed on load characteristics of equipment connected to the network grid to reduce damage to sensitive equipment from unbalanced power.
- *Proactive conduct is possible*: By implementing this research development, a power network with permanently installed power quality apparatus then permits proactive, rather than reactive management behaviour to be carried out.
- *Recent Digital Technology:* This research was built on previous studies in the area of digital technology, and it offers a framework for the evaluation and construction of future designs, as well as a variety of specific conclusions and recommendations.

1.5. The demarcation of the field of study

The objective of this research was to investigate and measure the contamination in the electric power network by using existing SCADA systems. Common sources of distortion in electric power systems, as well as their effects onto the power system were also investigated. This research can therefore be divided into five categories:

1) An investigation into the ESKOM North Western Regional electrical power system and the sources and consequences of power distortion that causes interference in the region.
2) Different types of power distortion and the specifications involved during this research.
3) The development of a Quality of Supply (QOS) power distortion measuring hardware and software module, referred to as the IED (Intelligent Electronic Device) to track down the main areas of concern in the Region.
4) The development of a relevant Database configuration at the Electricity Network Management and Control (ENMAC) system.
5) The integration and application of the above.

The flow diagram in figure 1.2 shows the steps followed during this development, and forms the basic layout of the thesis.

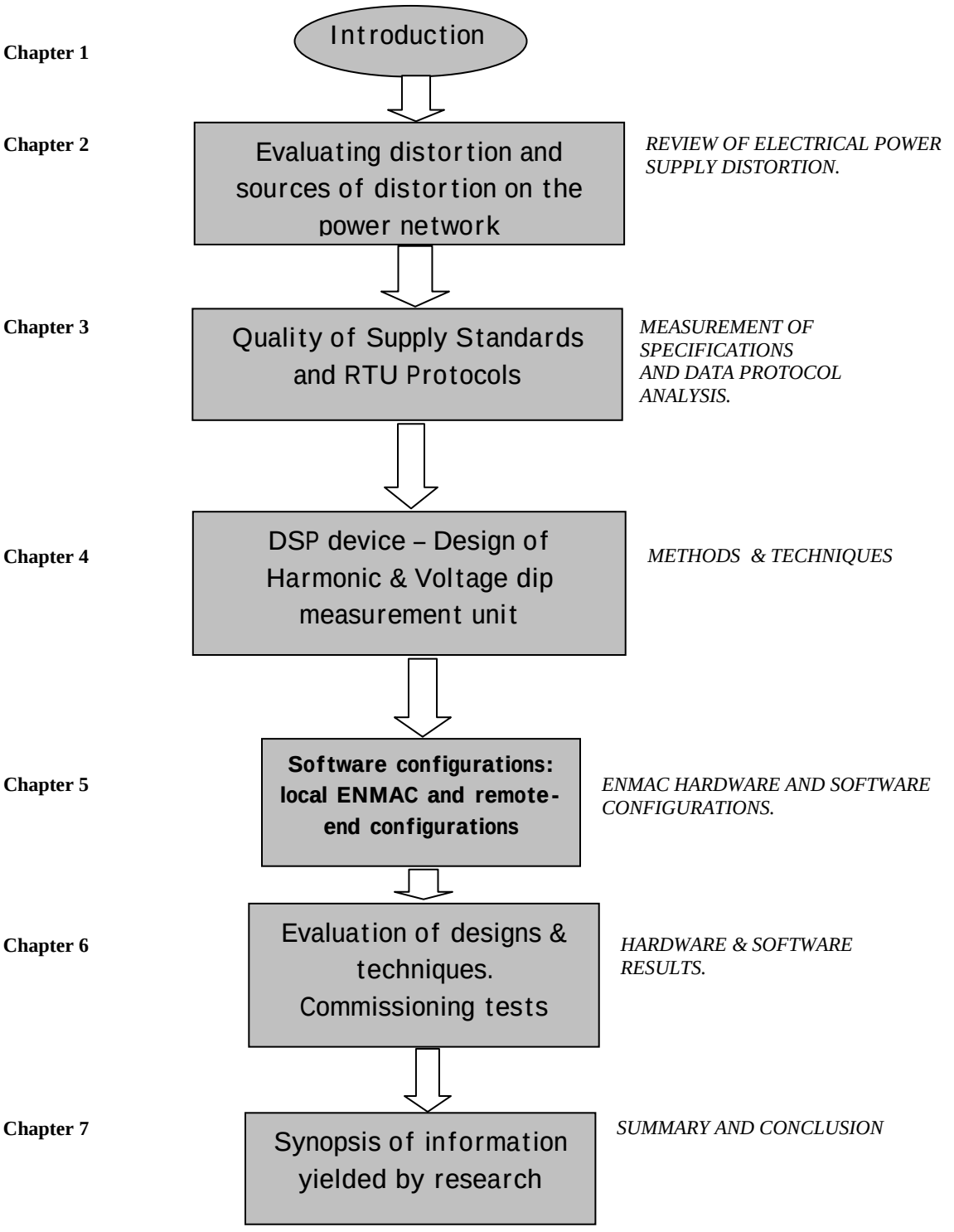


Fig. 1.2. Flow diagram of the phases followed during this research, involving the Remote Terminal Unit (RTU).

1.6. The prominence of this research

This research has the very practical intention to examine the most appropriate way to measure voltage distortion. At the time of writing, no suitable SCADA measurement unit, or any research on quality of supply measuring systems incorporating SCADA for use on power networks in South Africa (that could adapt to the existing Master station or function via available RTU's) exists. Concerning existing Telecontrol units in South Africa, and the protocols used at this point, this is unquestionably a pioneer assignment in the Eskom and South African environment.

1.7. Summary

Due to the influences and enhancement of modern electronic equipment, distortion and pollution on power networks occur more frequently. This can cause damage to equipment of both user and supplier facilities. Owing to the random phenomenon of disturbances, the need to constantly monitor the power network for such disturbances seems imperative.

Harmonic currents can have a significant impact on electrical distribution systems and the facilities that they feed. It is important to consider their impact when contemplating additions or changes to a system. In addition, identifying the size and location of non-linear loads and power pollution should be an important part of any maintenance, troubleshooting and repair program. This can be obtained by using a SCADA system.

This research can be divided into different categories, e.g. power distortion types and the development of measuring units and relevant software on both RTU and Central station equipment as anticipated by figure 1.2.

Because of systematic varying technology and techniques, this research was focussed mainly on contemporary technology SCADA units, working on DNP-3 protocol via radio, microwave or fiber-optic networks. For the time being, there is certainly no valid motivation why not to implement, and intensify this development and research on other Telecontrol RTU's and SCADA systems globally.

CHAPTER 2

REVIEW OF ELECTRICAL POWER SUPPLY DISTORTION AND DETECTION TECHNIQUES

2.1. Preface

In this chapter only selected detail on the relevant literature that was studied and consulted for the purpose of this research is provided. Because of the rapid changes in technology, great emphasis was placed on the latest findings and developments in the Power systems, signal processing as well as the Telecontrol protocol discipline.

This research document covers a wide field of problematical topics, integrated into one complete package to supply the optimum practical solution to assessing power pollution problems in South Africa. It was not possible to present a great deal of detailed information regarding distortion, integrated systems and protocol types due to the multifaceted nature of this research. Various and inspired literature on this subjects exist, and understandably, this thesis cannot attend to all the information and literature consulted.

2.2. Power systems

Most major power systems in South Africa normally operate in a balanced, three-phase sinusoidal 50Hz steady state mode, but certain situations like a very severe short circuit or "disturbance" can cause unbalanced network operation. When such an electrical disturbance occurs, usually one of the following effects takes place [13, p. 37]:

- End-user equipment can be disturbed or even damaged.
- A reduction in voltage occurs, attended with an increase in current flow.
- An arc is formed which, in extreme situations, can damage the equipment and also spread to other equipment, causing secondary damage.
- Switchgear can explode due to extreme disturbance situations, and can cause widespread damage.
- Oil on transformers and other plant equipment can ignite, due to heat.

On most power lines in South Africa, disturbances typically occur due to external causes such as fires, trees, birds and damage to cables or deterioration of the insulation. Short circuit current levels are an important part of many design projects, allowing the placement of protective devices, and designing a network to cope with increased fault levels [11, p. 7]. To protect the system against the effects of a disturbance, protection schemes are used to selectively disconnect the affected apparatus to allow the system to continue power transmission with the minimum amount of disturbance to both the supplier and consumer.

2.3. Power distortion

A major problem that most power utilities in South Africa have to face is to measure and control power system distortion accurately, in concurrence with the recommended standard - in this case the National Rationalised Standards (NRS-048) - without a major burden being placed on the price of electric energy. One of the finest definitions of Power Quality can be expressed as follows: “Any power problem manifested in voltage, current

or frequency deviations, that results in failure or miss-operation of customer operations" [27, p. 1].

The ideal situation is that when distortion is completely taken care of at the substation, there should be no secondary effects on other equipment coupled to the same feeder [30, p.27]. Previous experience demonstrated that power line disturbances usually are random phenomena, but when it occurs, it can be easily correlated with other in-plant happenings such as the switching of heavy loads [17, pp. 20, 34].

With the traditional test monitoring, the measuring period should be long enough (sometimes a few months) for every type of potential disturbance to be discovered. In many locations it may be necessary to monitor the power line through different seasons of the year, since disturbances may differ seasonally. Subsequently, the test equipment had to be transported back to office; data then had to be downloaded and processed. If no outcome was reached, the whole process had to start again, dissipating valued time and money. Disturbances like harmonics, voltage dips and other voltage conflicts could only be monitored via sophisticated measuring apparatus. Sometimes this equipment was semi-permanently installed at the substation, only to be moved to another measuring point when necessary. Unfortunately, one must submit that these tests usually depend on trial and error, and is a shoddy approach to solve this matter.

A realistic and practically orientated solution to support combating power contamination is therefore crucial, and this is not simple to achieve if one take into account the statement of Enslin, who accentuated that whatever you do, random line disturbances are the most difficult to combat and can **never** be eliminated completely [13, p. 13].

2.3.1. Voltage fluctuations

If one studies the field of Voltage surges, one finds that it is characterised by a temporary increase or decrease in voltage that may last for a few seconds. According to Stevenson [40, pp. 306-309] and Wadhwa [43, pp. 246-255], the most common cause of destructive voltage surges is an intermittently loose neutral connection and single line-to-ground (SLG) faults on other feeders causing a temporary voltage rise.

Wadhwa assured that switching off a large load or energising a large capacitor bank can also cause surges [43, p. 251]. This is because when a capacitor or unloaded feeder is switched, repeated re-strikes in the circuit breaker may produce over-voltages as shown in figure 2.1 [32, p. 281].

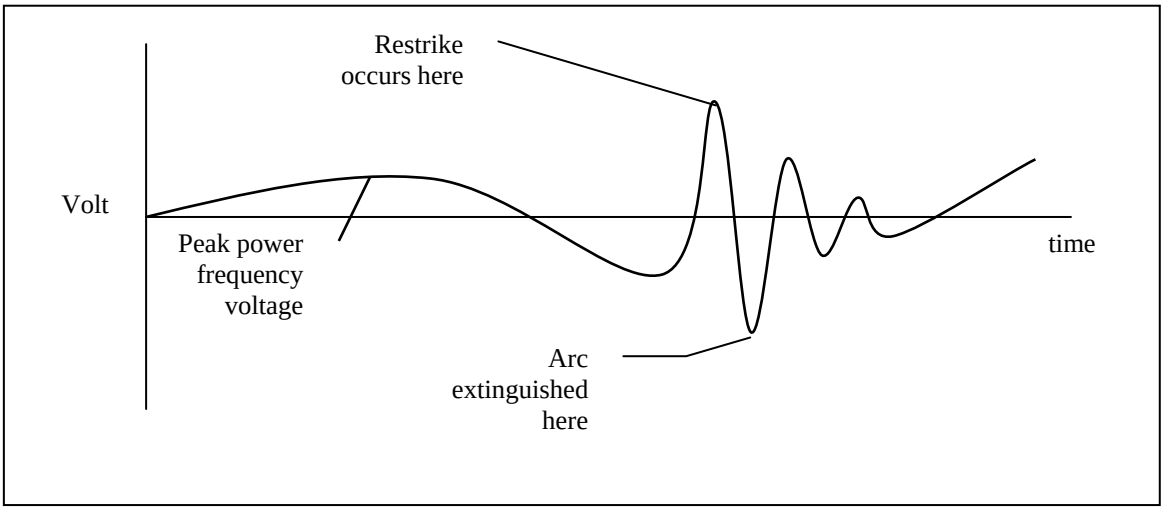


Fig. 2.1. Hypothetical phenomena as a result of strikes during the interruption of the capacity current of capacitors, unloaded cables or overhead lines [32, p. 281].

A voltage surge, the sudden change in voltage at a point in a system, can be from 16ms up to 3000ms in duration, usually with an increase of 10 – 35% above the normal line voltage [15, p. 235]. Consequently, surges and dips are short-term cases of over voltage

or under voltage conditions where the voltage fluctuation exceeds a pre-defined limit for at least some significant portion of a cycle, as shown in the following illustration.

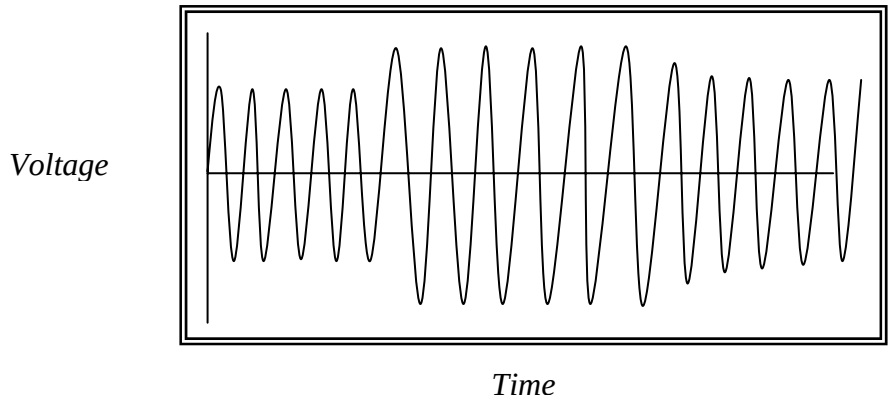


Fig. 2.2. Characteristic of a voltage surge condition [44, p. 61].

Because dips and surges are transitory conditions, which cause momentary occurrences, they usually exhibit larger voltage excursions than longer-term under-voltages and over-voltages [15, p. 236]. An alternative cause of transients is a surge resulting from a lightning stroke, which can be most destructive. In the case of lightning, surge-arrestors are used in most substations for protection purposes [15, p. 236].

Voltage fluctuations can also be the systematic variation of the voltage, or a series of random voltage changes. Loads, which can exhibit continuous, rapid variations in the load current magnitude, can cause voltage variations that are often referred to as "flicker" [26, p. 16].

Electronic equipment is especially vulnerable to damage because high voltage transients can very easily impregnate integrated circuits, if the integrated circuit was not completely destroyed after the transient. Transients can also be caused by switching with excessive arcing, such as motor-starter contactors with dirty or badly aligned

contacts - thus the IEE recommends that all sensitive electronics should be protected with Transient Voltage Surge Suppressors [21, p. 178].

2.3.2. Voltage dips

Voltage dips (sometimes called brownouts) are a sudden reduction or decrease of the normal supply voltage, for a short period of time of between a few milliseconds up to 30s, [44, p. 61]. Voltage dips are mainly caused by power system faults that can vary from short circuits on the power system to the starting of large motors [44, p. 61].

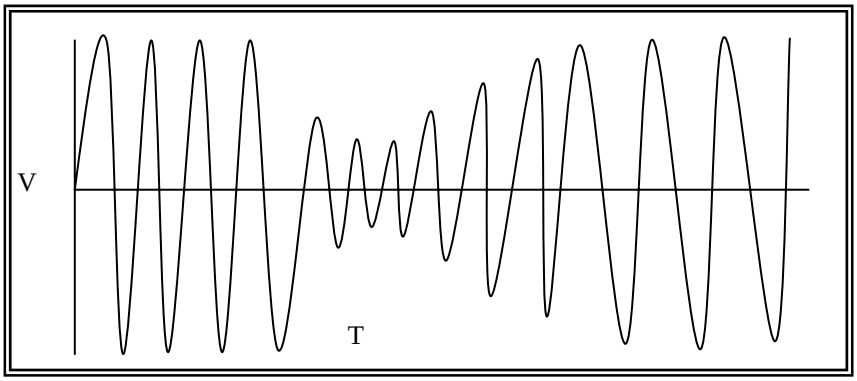


Fig. 2.3. Typical example of a voltage dip condition [44, p. 61].

A significant concern of electrical distributors is that motors on industrial power systems are becoming increasingly larger, and can cause severe disturbances to any locally connected loads [21, p. 140].

Thus far it is known that Voltage dips are typically caused by fault conditions, but motor starting can also result in under-voltages. These are typically longer in duration than 60 cycles and the associated voltage magnitudes are not as severe [33, p. 21]. Motor

starting voltage variations are often referred to as "voltage flicker", especially if the motor starting occurs frequently [33, p. 22].

Figure 2.4 shows a simulation of a typical voltage dip caused by a motor starting.

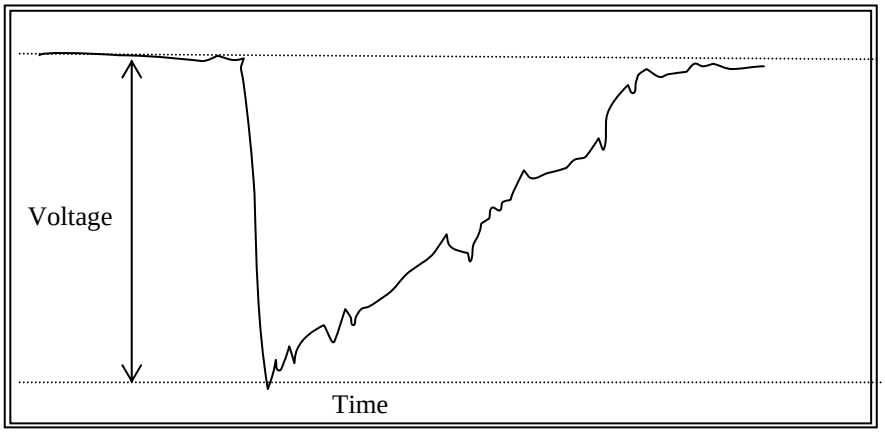


Fig. 2.4. Typical example of a voltage dip caused by a starting motor [21, p. 143].

Studying historic data records and fault complaints in the Regional Control centre in Bloemfontein, the author observed that voltage dips are one of the most important power quality problems facing our rural consumers these days. Countless complaints are reaching power utilities, especially when irrigation systems fail due to relays and contactors in motor starters that are sensitive to voltage dips, resulting in the shut down of a process. According to the National Electricity Regulator (NER), these types of voltage dips can also be described as momentary decreases in the RMS voltage magnitude, and are usually caused by a remote fault somewhere on the power system [28, p. 9]. The National Electricity Regulator (NER) also proclaims that all dips larger than 10% of the RMS voltage can legitimately be considered as a “dip” [28, p. 9].

Also, certain fault conditions, such as insulation flashover, and the connection of large loads usually cause extreme voltage dips [3, p. 11], and this condition could last until a protective device clears the fault. In the power substation, this will typically be a circuit breaker [44, p. 62].

Although most utilities go to great lengths to prevent faults on the system, they cannot be eliminated completely, because usually these faults are temporary, which means that they will not reinitiate after they have been cleared and the line is reclosed [12, p. 333]. Since faults (and, therefore, voltage dips) are inevitable [32, pp. 294, 302], it is important for customers to make sure that critical equipment sensitive to voltage dips is adequately protected [32, pp. 164,165]. A large majority of faults on a utility system are Single Line-to-Ground Faults (SLGF). Gross stated that SLGF's often result from weather conditions such as lightning and wind [16, p. 269]. Three-phase faults are more severe, but much less common [27, pp. 344-347].

2.4. Equipment sensitivity to voltage dips

Industrial equipment can be particularly vulnerable to voltage dips because the equipment is often interconnected, and a trip of any component in the process can cause the whole plant to shut down. Examples of these include [43, pp. 85-97]:

- Paper industries.
- Plastics and petrochemical industry.
- Textile industries.
- Semiconductor manufacturers.

- Rubber industries.

Control devices such as computers, contactors, and programmable logic controllers are often supplied through a single-phase control transformer. The voltages experienced during a voltage dip condition will depend on the equipment connection. One would imagine that voltage unbalance would be a significant matter for motor heating, especially for long duration unbalances, but Gross considers the duration of the unbalanced voltages during fault conditions to be so short that motor heating is not a significant concern [16, pp. 259, 267].

Adjustable speed drives, nevertheless, may have controls that trip very quickly during unbalanced conditions. To wrap up, so far it seems that different categories of equipment, and even different brands of equipment within a category, have significantly different sensitivities to voltage dips.

2.5. Ferromagnetic devices

Ferromagnetic devices can be categorised under devices with iron cores, with coils wounded around it like transformers and motors [10, p. 7]. If we study the T-equivalent model of these devices, it shows that the series impedance is linear, while the magnetising impedance is disgustingly non-linear [14, p. 702].

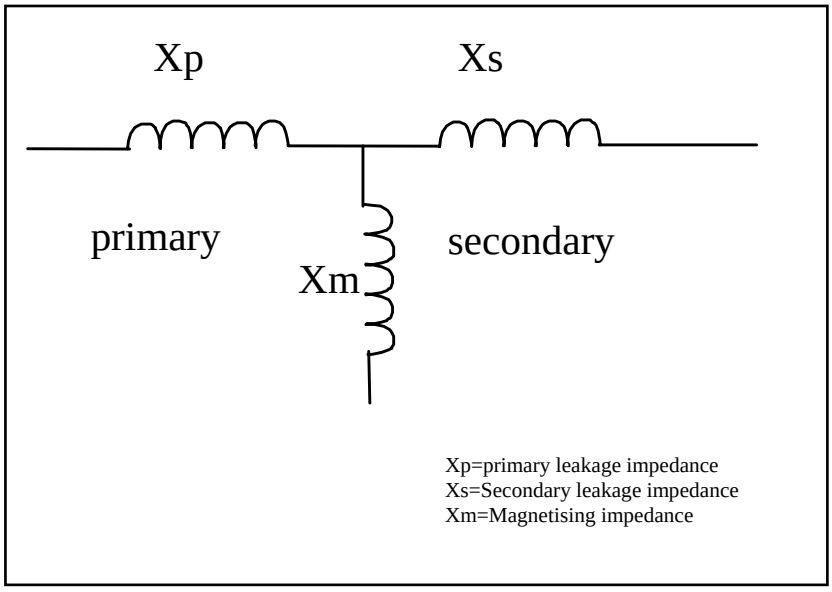


Fig. 2.5. T-equivalent model of a transformer [10, p. 7].

In the book "Electrical Distribution Energy" by Pansini [31, pp. 81-82], it is said that for many reasons, including the random and non-uniform movement of molecules in the cores of a transformer, the alternating magnetic field that is set up may be distorted. This can produce serrated or jagged sine waves on both sides of the transformer [31, p. 82], which can be broken down into a series of harmonics with frequencies of 3, 5, 7, etc., times the fundamental frequency. He also stated that if the transformers have a ground on either side, the harmonics will flow to ground, and the original sine wave remains undistorted [31, p. 82]. In the book "Electric Power System Harmonic Design Guide", Dugan and McGranaghan claim that in such an instance, if no ground exists, the triplen harmonics can be bothersome [10, p. 7]. This can be conquered by the delta windings, used to circulate the harmonics within it [10, p. 7], so that little or none is transferred to the primary windings. This may produce a little heat, but it will filter out the harmonics [10, pp. 7-9].

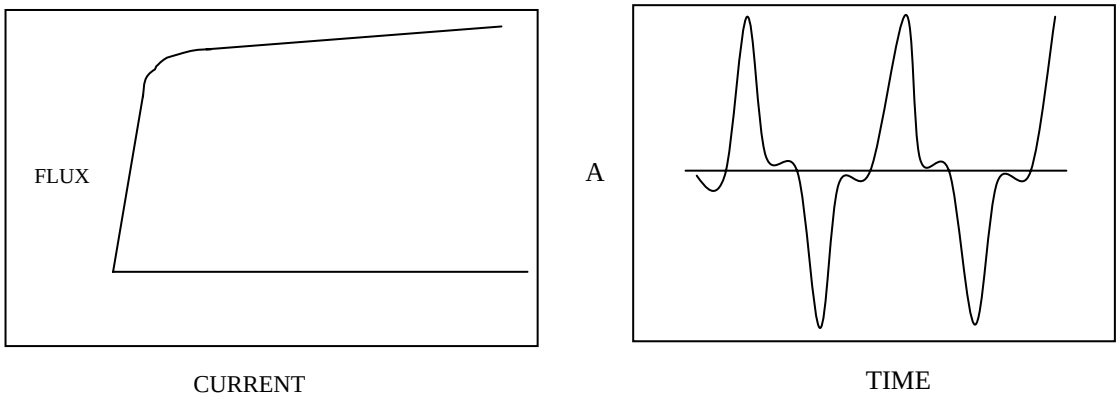


Fig. 2.6. Example of a flux-current relationship and magnetising current wave-shape of a transformer [10, p. 7].

According to Franklin [14, p. 654], third harmonic current can overheat a transformer, but the only case where it is likely to become serious in practice is where the transformer primary windings are connected in interconnected star, with the generating feeder and transformer neutrals being joined together. Figure 2.6 shows the typical flux-current relationship curve, as well as the transformer magnetising wave-shape.

While investigating complaints during 1995 on telephone line disturbances caused by harmonics on nearby ESKOM electrical lines, the author found that the major cause of these disturbances was discriminative protection gear generating third harmonic current components. There the author also found that third harmonic voltages can charge adjacent lines and telephone cables electro-statically, and this was an irritation to a lot of users on that line. It was found that the harmonics on the mains supply caused disturbances on the entire telephone line, causing discomfort to hundreds of telephone users. Whitaker mentioned that this type of disturbance only occurs on a star connection, when the fourth wire is used as one of the cable cores or the earth [44, p. 150], as was the case on that specific line. These third harmonic voltages can increase transformer insulation stress, which will categorically have an effect on reliability and the efficiency of the transformer,

causing losses, i.e. copper losses in both windings, and iron losses in the core due to hysteresis and eddy-currents [20, p. 410].

Even if voltages between lines remain normal, the circuit can raise to an indefinite potential above ground if adjacent currents are not grounded properly [14, p. 656]. This causes the insulation to ground to become unduly stressed, and the life of apparatus is therefore reduced [14, p. 657]. Also worth mentioning is that possible resonance of transformer windings and line capacitance can also occur due to third harmonic voltages [21, p. 166].

In an AC arc furnace, the arc is non-linear, asymmetric and unstable and generates a spectrum including odd and even harmonics as well as a continuous spectrum as shown in figure 2.7. The spectrum depends on the type of furnace, its power rating and the operation considered (e.g. melting, refining). Measurements are therefore required to determine the exact spectrum. In DC arc furnaces the arc is supplied via a rectifier and is more stable than the arc in AC furnaces. According to Dugan and McGranaghan [10, p. 10], most arcing devices - this includes fluorescent lighting, sodium and mercury vapour lighting and arc furnaces - have similar current and harmonic spectra as shown in figure 2.7.

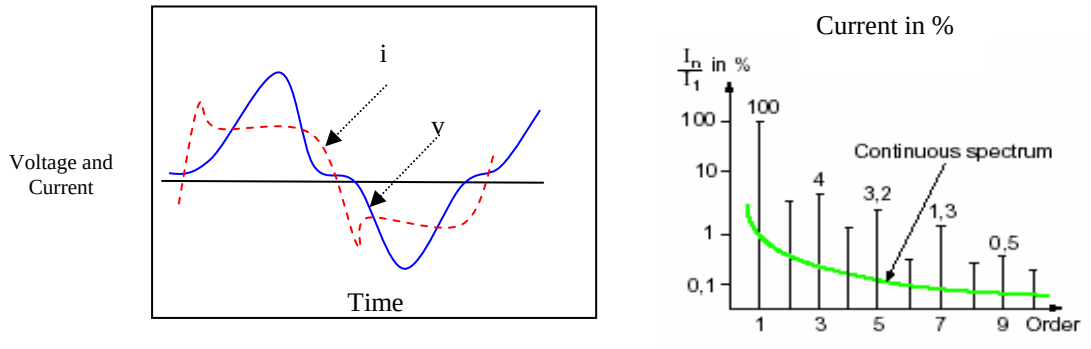


Fig. 2.7. Typical arc-furnace voltage and current waveform, as well as current spectrum for an arc furnace [10, p. 10].

2.6. Harmonic distortion in power systems

If one considers the problems caused by harmonics, Enslin recapitulates that most potential problems caused by harmonics in South Africa are [13, pp. 56-61]:

- Unpredictable equipment operation.
- Overheated transformers.
- Rise in temperature and increase in power losses on rotating machines.

Rectification and phase-angle control, while essential to the design of modern equipment, are also major causes of harmonic currents [1, pp. 211-213, 323]. In fact, any device that has a non-linear voltage-current or load characteristic causes harmonics, such as variable speed drives, electronic rectifiers, power supplies and arc furnaces [10, p. 111]. It must be emphasised that these harmonic currents can cause excessive heating and failure of wiring, connections, transformers and capacitors, and the measurement and detection thereof was prominent during this research in an attempt to reduce that phenomenon.

We can observe that waveforms of voltage and current, which are not correlated (like two sinusoids of different frequencies), result in a loading of the power network without transferring net energy [13, p. 40]. If the correlation between the two signals decreases, the net amount of energy also decreases, although the load may remain constant, and this implies an increase in distortion [13, p. 41]. While the harmonic-producing equipment is a load for the fundamental current, it can be viewed as a source of harmonic current. The level of harmonic current flowing into the system impedance determines the harmonic voltage distortion level [1, p. 219]. Since the system impedance is usually low, the magnitude of voltage harmonics, and the extent of voltage distortion, is usually lower than that for the corresponding current distortion.

Quite a lot of substations have capacitor banks installed. The effect of these capacitors, such as those used for power factor correction, can result in local system resonances [10, pp. 38, 111, 166]. Industrial clients are then often required to limit the disturbances they cause on the utility grid and to compensate their power factor, unbalance and harmonic levels to avoid billing penalties [3, p. 1]. However, with all that in mind, care must be taken that no critical resonance frequency is introduced and that harmonics present in the system are not excessively amplified as resonance can occur.

Harmonics can also sometimes be more of a local problem for the end user than for the supplier, especially when compared to voltage dips and interruptions, which represent most power quality problems. Another informative fact to remember is that harmonic currents due to non-linear loads usually flow from the customer source towards the utility, as shown by figure 2.8 [10, p. 13].

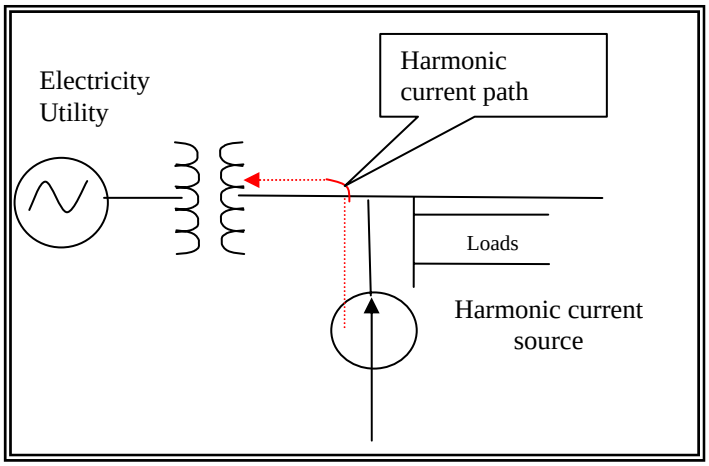


Fig. 2.8. Example of harmonic flow in a distribution system [10, p. 13].

2.7. Possible ways of determining the existence of harmonics

Taking into consideration all the information so far, it should not be very complex to determine possible harmonic presence at a substation plant by immediate observation methods like:

- Are any transformers hot and noisy?
- Are neutral conductors extremely hot and carrying too much current?
- Do induction motors within the facility fail frequently or run extremely hot?
- Do output filter capacitors of uninterruptible power supplies (UPS) fail frequently?
- Do standby power generators operate poorly?

2.8. Calculation of harmonic frequency

A pure sinusoidal waveform can be expressed as a function of time, $I=i_m\sin(2\pi f t)= i_m\sin(\omega t)$, completing f cycles in 1 s with a period $T=1/f$ [26, p. 3/10], and the graph of a sinusoidal current or voltage of frequency f can be represented by the projection of a line

of angular speed ω about one end [26, p. 3/10]. A stationary line can represent the sine wave, particularly in relation to other sine waves of the same frequency, but "out of step". Two such waves, say *v* and *i*, with peak values v_m and i_m respectively can then be written as $v=v_m\sin\omega t$ and $I=i_m\sin(\omega t-\phi)$; and drawn as in figure 2.9 [26, p. 3/10].

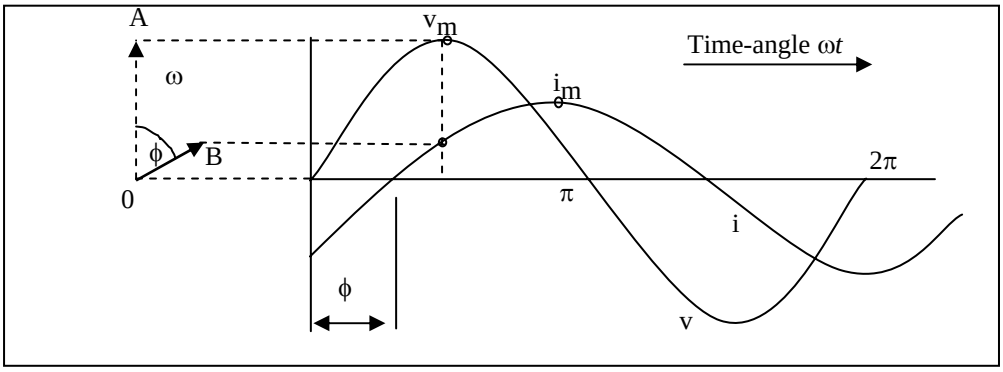


Fig. 2.9. Phasor diagram showing the phase angle between 2 waves v and i rotating counter clockwise at angular speed ω [26, p. 3/10].

Where:

v and i represent the two waves

OA and OB represent the two waves in peak magnitude and relative time phase

v_m – peak value of *v*

i_m – peak value of *I*

ω - angular speed

ϕ - phase difference or phase angle between *v* and *i* in rad.

According to the Electrical Engineer's Reference Book, any uni-valued periodic waveform can be represented as a summation of sine waves comprising a fundamental, where frequency is that of the periodic occurrence, and a series of harmonic waves of frequency 2, 3, 4 …, n times that of the fundamental [26, p. 1/8]. Thus, any periodic wave can be described mathematically as a series of sinusoids summed together. This is known

as the Fourier series, which for a periodic function y = f(x) takes either of the following equivalent forms [26, p. 3/13].

1) $y = c_0+c_1 \sin(x+\alpha_1) + c_2 \sin(2x+\alpha_2) + \ldots$ ----------(2.1)

or

2) $y = c_0+a_1 \cos x + a_2 \cos 2x + \ldots a_n \cos nx +$

$b_1 \sin x + b_2 \sin 2x + \ldots + b_n \sin nx$ ----------(2.2)

Where c_0 = constant,

$c_n = \sqrt{(a_n^{\,2} + b_n^{\,2})}$ *and*

$\alpha_n = \arctan(a_n / b_n)$

The coefficients of the terms are given by:

$c_o = (\frac{1}{2}\pi)\int_0^{2\pi} f(x)\,\mathbf{d}x$ = mean of the wave over one period,

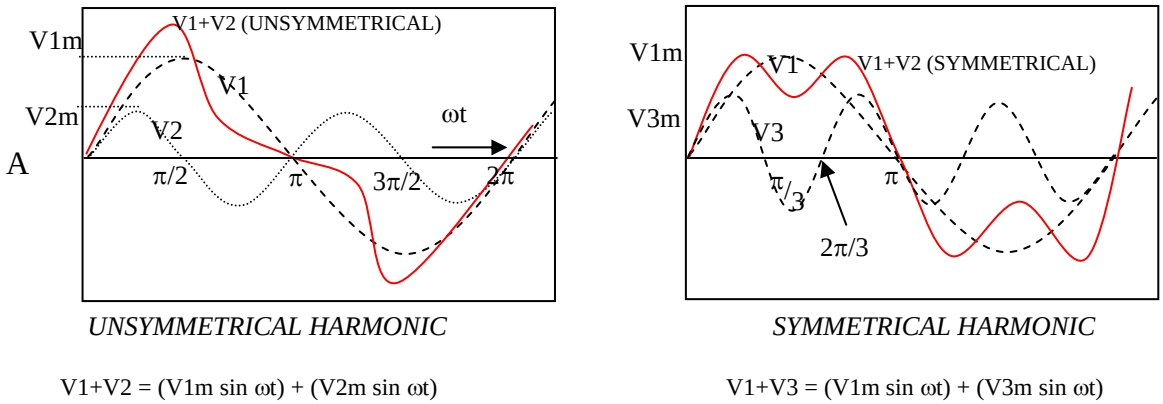
$a_n = \left(\frac{1}{x}\right)\int_0^{2\pi} f(x)\cos n_x dx$, *and* $b_n = \left(\frac{1}{x}\right)\int_0^{2\pi} f(x)\sin n_x dx$

These can be evaluated mathematically for simple cases [26, p. 3/13].

The sinusoids comprising harmonics are integer multiples of the frequency represented by the fundamental periodic cycle. Each term in the series is referred to as a “Harmonic” of the fundamental frequency [9, p. 147].

If it has the same frequency as the fundamental, it is called the first Harmonic, and if the term has twice the fundamental frequency, it is called the second harmonic, etc. Symmetrical waves contain only odd harmonics, while unsymmetrical waves contain the even harmonics [10, p. 2].

A sample of such waveforms is shown in figure 2.10.



UNSYMMETRICAL HARMONIC

V1+V2 = (V1m sin ωt) + (V2m sin ωt)

SYMMETRICAL HARMONIC

V1+V3 = (V1m sin ωt) + (V3m sin ωt)

Fig. 2.10. Waveforms with unsymmetrical and symmetrical harmonic components [10, p. 2).

One of the waves that are shown repeatedly in harmonic analysis is the square wave resulting from rectifiers or arc furnaces [21, p. 165]. The Fourier series for such a wave is shown as [10, p. 3]:

$$V(t) = 4V/\pi * (\sin\omega t + 1/3 \sin 3\omega t + 1/5 \sin 5\omega t + 1/7 \sin 7\omega t + \ldots) \text{----------} (2.3)$$

A non-periodic signal is called “aperiodic”. A discrete signal is called a sampled signal [6, p. 3], and a Fourier series can represent both non-periodic and discrete signals [6, p. 6]. Extracting useful information and discarding the extraneous information from measured data, which can be mathematically represented as a scalar or vector function of one or more variables constituting the index set, are defined as signal processing [6, p. 3].

We also see that exponentials are related to sinusoidal signals of the form $x(t)$ = A Cos $(\Omega_k t + \varphi)$, where both Ω and φ are in units of radians [6, p. 4]. The signal and its Fourier series representation are shown in figure 2.11, that also show the coefficients over one period.

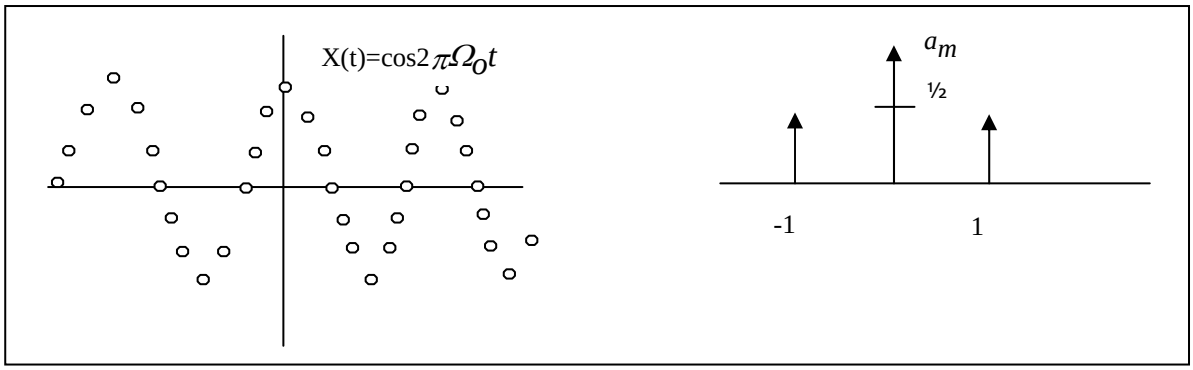


Fig. 2.11. Discrete values describing a sinusoid and showing the function and the spectral coefficients respectively [6, p. 7].

James stated [25, p. 6] that in order to determine the relative distortion due to harmonics on a power system, the term Total Harmonic Distortion (THD) has emerged. Total Harmonic Distortion is a measure of the amount of distortion harmonics cause on the system voltage, expressed as a percentage of the fundamental [25, p. 6]. Both voltage and current waveform distortion may be represented by THD, with Total Harmonic Voltage Distortion (THVD) and Total Harmonic Current Distortion (THCD) sometimes used to distinguish between the two [25, p. 6]:

$$THD = \sqrt{\frac{Sum\_of\_squares\_of\_amplitudes\_of\_all\_harmonics}{Square\_of\_amplitude\_of\_fundamental}} x100\% \text{ ----------(2.4)}$$

$$THD = \frac{\sqrt{\sum_{h=2}^{\infty} V_h^2}}{V_1} X100\%$$

where:

Vh is the RMS value of the harmonic component h.

V1 is the RMS value of the fundamental component.

2.9. The Fourier transform

The Fourier transform is widely used in solving problems in science and engineering [5, pp. 2-3], and has also been very successful in the restoration of astronomical data [4, p. 448]. The Fourier transform is used widely in linear system analysis as a very effective tool to alter a problem into one that can be more easily solved [5, p. 3]. The Fourier transform, in essence, decomposes or separates a waveform or function into sinusoids of different frequencies that sum to the original waveform. It also identifies or distinguishes the different sinusoids and their respective amplitudes [5, p. 4].

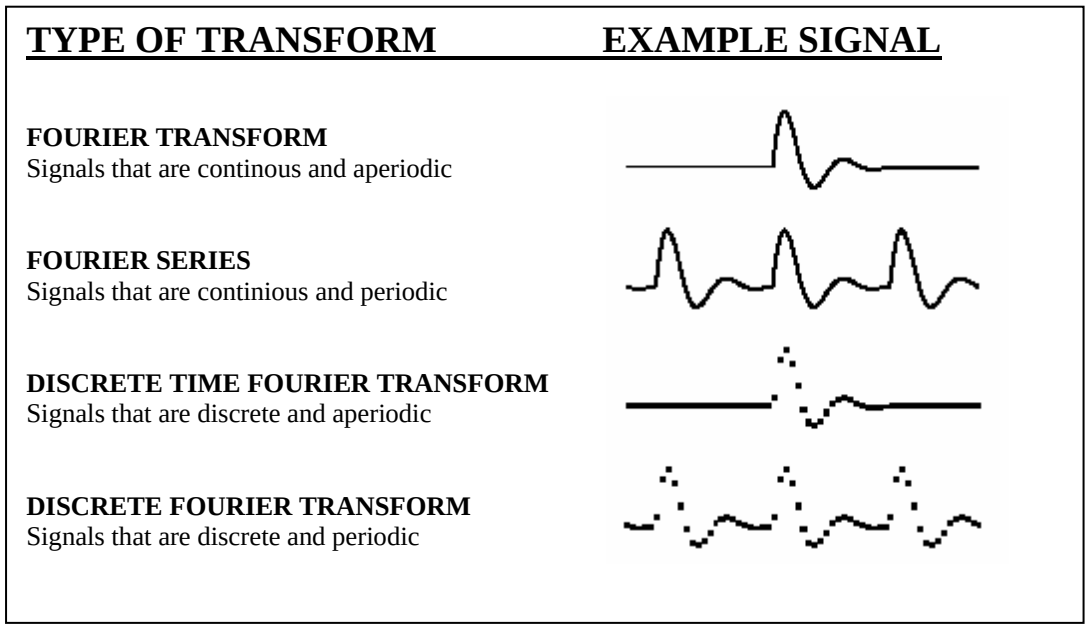


Fig. 2.12. Illustration of the four Fourier transforms [5, p. 37] and [38, p. 145].

The discrete Fast Fourier transform (FFT) could also be used to evaluate the frequency response of discrete-time systems, and is an algorithm that converts a sampled complex-valued function of time into a sampled complex-valued function of frequency [38, p. 139].

The FFT tends to be better suited to analysing digital audio recordings than for filtering or synthesising sounds [22, p. 209], e.g. to do the software equivalent of a spectrum analyser used for displaying a graph of the frequency content of an electrical signal.

2.10. The Goertzel algorithm

Countless applications, including this research, require the detection of only a few discrete sinusoids. According to the members of the Siglab team [29, p. 1], the Goertzel filter is an Infinite Impulse Response (IIR) filter (digital filter for DSP to achieve a given filtering characteristic - using less memory than a similar Finite Impulse Response (FIR) filter) that uses feedback to generate a very high-Q bandpass filter where the coefficients are easily generated from the required centre frequency.

The most common configuration for using this technique is to measure the signal energy before and after the filter and to compare the two [29, p. 1]. If the energies are similar then the input signal is centred in the pass-band, if the output energy is significantly lower than the input energy then the signal is outside the pass-band [29, p. 1].

The Goertzel algorithm is most commonly implemented as a second order recursive IIR filter [29, p. 1], and is defined by the following equation:

$$Hf(z) = \frac{1 - \frac{2\pi f_i}{e f_s Z^{-1}}}{1 - 2\cos\left(\frac{2\pi f_i}{f_s}\right) z^{-1} + z^{-2}} \qquad \text{----------(2.5)}$$

Where fi is the frequency of interest, and

fs is the sampling frequency.

Outlines of a second order recursive Goertzel filter is shown in figure 2.13.

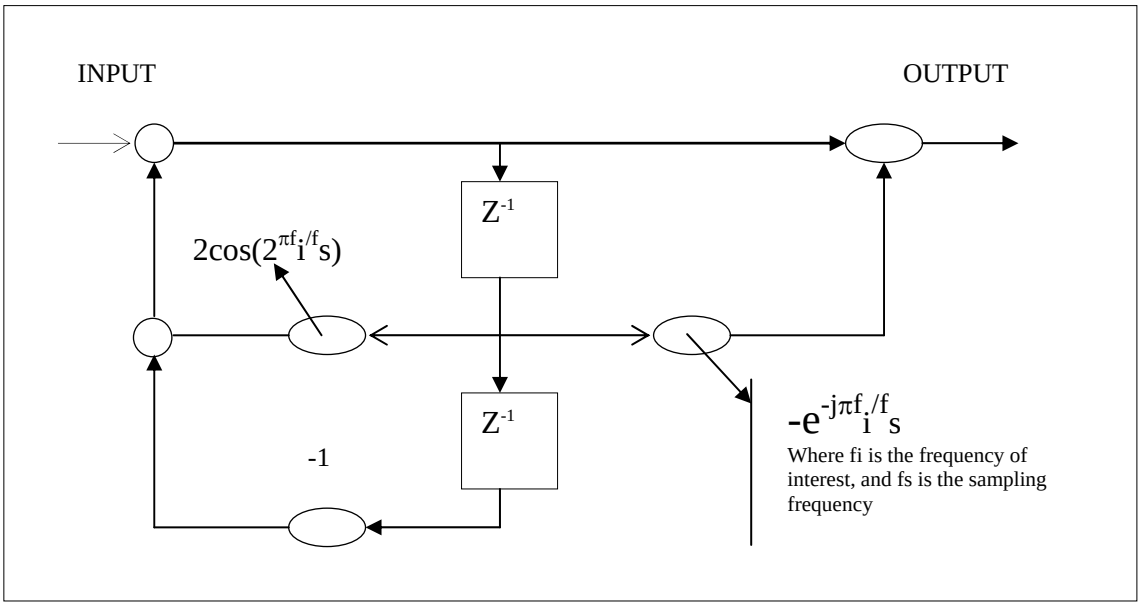


Fig. 2.13. The Goertzel algorithm used as a second order recursive IIR filter [29, p. 2].

2.11. Summary

Wide-ranging sources of power pollution and its effect on the electrical power system generally arises because of the distorted, fluctuating, or unbalanced currents drawn by utility customers. These interact with the utility network, giving rise to degradation of the quality of the voltage supplied to a neighbouring customer. It is this "pollution" of the system voltage that can potentially affect the equipment operated by other customers connected to the utility network.

To an ever-increasing extent, modern industrial processes require high-quality electricity supply. It is becoming essential to be independent of unforeseen fluctuations inherent in the transmission of electricity. Because many types of equipment do not draw sinusoidal currents, this "pollution" of the voltage cannot be circumvented, and as a result utilities

should design and operate their networks to achieve acceptable levels of pollution (voltage quality) that are adequate to most items of utility equipment. Likewise, equipment designers need to design equipment robust enough to withstand acceptable levels of "pollution" of the supply voltage. These "acceptable" levels have been defined by the NER.

Single-line-to-ground faults on the utility distribution or transmission system are often the cause of voltage dips, and lightning is a frequent cause. Subsequently, voltage dips, surges and harmonics are the three most devastating sources to be dealt with during this research. Because many types of equipment do not draw sinusoidal currents, this "pollution" of the voltage cannot be circumvented, and as a result utilities should design and operate their networks to achieve acceptable levels of pollution (voltage quality) that are adequate to most items of utility equipment.

If we look at the evidence it is very clear that voltage dips and harmonic distortion turn out to be an escalating concern for process industries due to growing automation of plants and facilities. Automated facilities are more difficult to restart, and the electronic controllers that are used are more sensitive to voltage dips than other loads.

Although interruptions and outages can be monitored and confirmed, it is imperative to emphasise that only through customer co-operation and involvement, can these solutions be successfully implemented for the prevention of outages, plant damage and production loss.

CHAPTER 3

MEASUREMENT SPECIFICATIONS AND DATA PROTOCOL ANALYSIS

3.1. Introduction

This chapter deals with the approved standards used for measuring the quality of electricity supply, as well as the data protocols used for measuring it. It was also during this phase of the research that the author realised that some dissimilarity exists between different suppliers and products on software techniques, especially with the practical implementation of the DNP-03 protocol. This obstruction was ultimately conquered by focussing on the original Harris DNP-03 standards.

The information presented here is sufficient to familiarise the reader with the most significant technological terms, applications and operations used during this research.

3.2. Standards for measuring electricity supply

With the growth of industry and technology, the availability and the quality of the electrical supply considerably improved to a very high level during the last few years. But valuable clients typically expect availability of electricity to be 100%, and this kind of request is complicated to meet in practice and at the end, someone has to pay for it. Solutions to ensure that suppliers and users of electricity stay compatible with each other therefore had to be initiated. International standards addressing the issue of compatibility started evolving and are reaching a mature level at present.

The European market is using the European norm - EN50160, and the new IEC standard IEC61000-4-30 as tools to balance the need for compatibility while the new IEEE 5960 standard is used in the USA [21, p. 62]. Most of the international standards are not addressed in isolation and are evolving to become one international power quality standard [28, p. 4].

However, in some countries like South Africa, where the electricity supply market is not open to competition yet, the same need for compatibility also exists. They usually establish national energy regulators that can act as a system watchdog with authority to settle any disputes between suppliers and users of electricity. In South Africa the National Electricity Regulator (NER) developed regulations published as the NER directive on power quality, a regulatory framework for the management of power quality in South Africa, to regulate the market [28, p. 2].

The monitoring of power quality has therefore become a very essential aspect of the day-to-day activities of any modern electricity supplier and consumer. The NER was aware of the urgent need to provide standards within the electricity supply industry to regulate the quality of power supply. The necessity to provide standards for the quality of service to the customers of electricity distributors was equally obvious. Therefore the National Regulatory Standards (NRS) NRS-048 was accepted. A working group, comprising of members drawn from a wide cross-section of the electricity industry, was established to prepare the document under the guidance of the NER.

3.2.1. National Regulatory standards for voltage dips

The NER standards for voltage dips are characterised by the measurement of the dip duration below the dip threshold, and by the dip magnitude as shown in figure 3.1. The duration of a voltage dip according to the NER [28, p. 9], is the time measured from the moment the R.M.S. voltage drops below 90% of acknowledged voltage, to when it rises above 90% of acknowledged voltage.

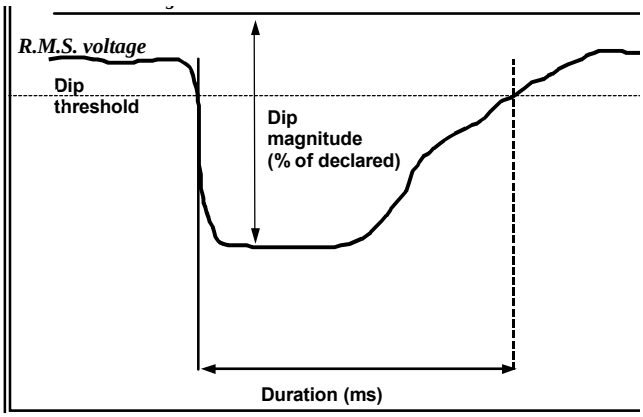


Fig. 3.1. Graph showing the measured voltage dip parameters [28, p. 9].

According to the NER, the duration of a dip is measured in milli-seconds, and is expressed in time frames, e.g. [28, pp. 8, 9]:

- Dip duration of 20-150ms is classified as a type "**X**" dip.
- Dip duration of 150-600ms is classified as a type "**S**" dip.
- Dip duration of 600-3000ms is classified as a type "**Z**" dip.

The magnitudes of voltage dips are measured in percentage of the nominal voltage, e.g.:

- A <10% dip is not regarded as a dip, therefore recorded as such.

- A 10-20% dip is classified as a type “Y” dip if within a specified dip duration as explained above.
- A 20-60% dip is classified as an “X”, “S” or “Z” dip; depending on the dip duration.
- A >60% dip is classified as a “T” dip; depending on the dip duration.

Combining the dip duration and magnitude, voltage dips can be represented graphically, in terms of duration and magnitude, on a graph known as a voltage dip window as shown in figure 3.2.

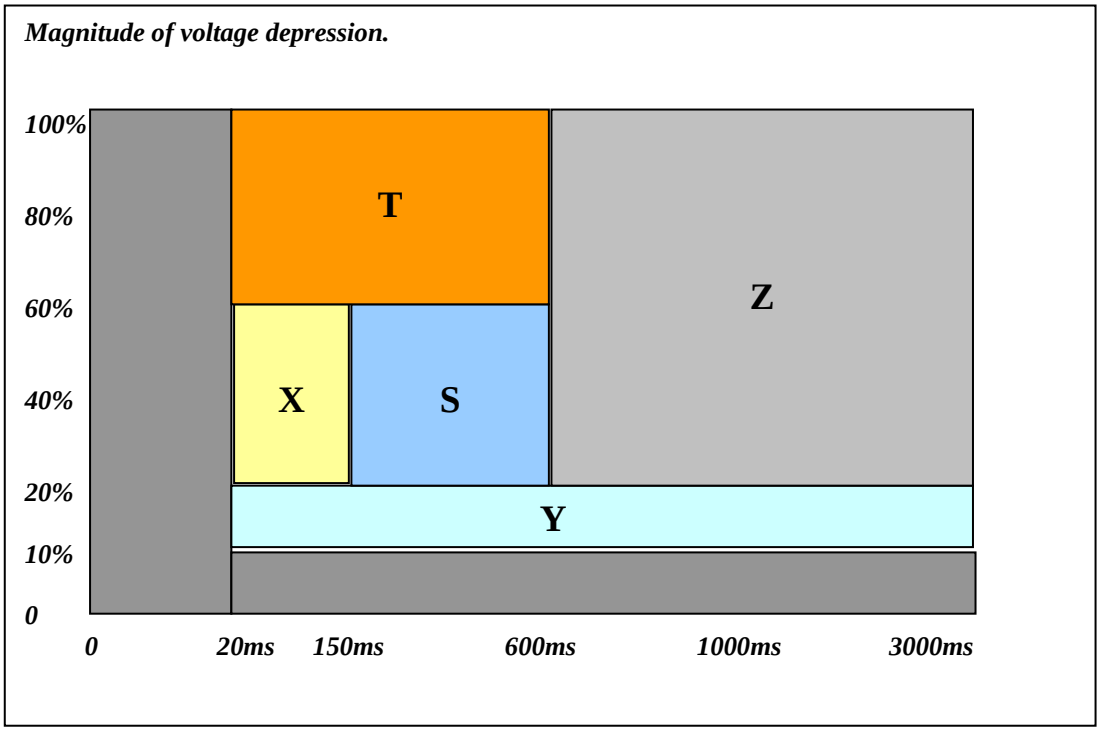


Fig. 3.2. The voltage dip window [28, p. 10].

3.2.2. Compatibility levels for the voltage dip window

The NRS compatibility voltage dip levels are given in the form of a maximum number of dips per year for defined ranges of voltage dip duration and magnitude, designated as

dip window categories “S”, “T”, “X”, “Y”, and “Z” [28, p. 10]. This is shown in figure 3.2. The measurement accuracy in the duration of the voltage dip should be 10ms, while the total accuracy in the minimum dip magnitude should be ± 2,5% of the nominal voltage [28, p. 10]. The compatibility levels set out for voltage deviation by the NRS for voltages above 500V is given as ± 5%, while voltages below 500V is set out as ± 10% [28, pp. 9-11].

3.2.3. National Regulatory standards for harmonics

The Rationalised User Specification for Harmonic measurement according to the NRS is as follows:

- All three phases of the supply voltage shall be monitored [35, p. 8].
- In the case of systems with solidly earthed transformer neutrals, the phase-to-earth voltages shall be measured [35, p. 8].
- In the case of delta-connected systems or systems with impedance earthing or which are unearthed, the phase-to-phase voltages shall be monitored [35, p. 8].
- The assessment period shall be at least 7 continuous days [35, p. 8].
- On each phase of the supply voltage, the instrument samples and records each harmonic voltage at intervals of 3 s or less [35, p. 8].
- These samples are summated over each 10-minute period, to obtain 10-minute root-mean-square values over each period [35, p. 9].
- For each harmonic and for the Total Harmonic Distortion (THD), for each 24 h day (00:00 to 24:00), the highest 10 min root-mean-square (RMS) values which are not exceeded for 95 % of the time are recorded for each phase [35, p. 9].

- For each harmonic and for the THD, the highest of the values on each phase shall be retained as the assessed daily values [35, p. 9].

The total harmonic distortion (THD) of a signal is the ratio of (a) the sum of the powers of all harmonic frequencies above the fundamental frequency to (b) the power of the fundamental frequency. For example, for currents, the THD is defined as [37, p. 10]:

Total Harmonic Distortion (THD) = IH/IF ----------(3.1)

Where:

$$I_H = \sqrt{I_2^2 + I_3^2 + ... + I_n^2}$$

I_n : RMS value of the harmonic n

I_F : RMS value of the fundamental current

The number of days during the assessment period that the THD level exceeded the compatibility level given in table 3.1 must also be recorded [35, p. 8].

Table 3.1. NER compatibility levels for harmonic voltages (expressed as a percentage of the declared voltage of HV and EHV systems) [35, p. 14].

1	2	3	4	5	6
Odd harmonics non-multiple of 3		Odd harmonics multiple of 3		Even harmonics	
Order h	Harmonic voltage %	Order h	Harmonic voltage %	Order h	Harmonic voltage %
5	6	3	5	2	2
7	5	9	1,5	4	1
11	3,5	15	0,3	6	0,5
13	3	21	0,2	8	0,5
17	2	>21	0,2	10	0,5
19	1,5			12	0,2
23	1,5			>12	0,2
25	1,5				
>25	0,2 + 1,3 × 25/h				
Total harmonic distortion (THD) ≤ 8 %					
NOTE — For each harmonic, the harmonic voltage distortion compatibility level is given as a percentage of the magnitude of the declared (fundamental frequency) voltage.					

Table 3.1 above shows the maximum allowed NER levels for harmonic voltage on High Voltage (HV) and Extra High Voltage (EHV) systems, as used in South African power systems. By implementing the outcome of this research, all power accessories can now be monitored and compared not to exceed the levels as indicated by the NER.

3.3. SCADA Remote and Master unit protocols

While working on this research, the author established that the designers of communication protocols usually do not provide formal proofs of correctness of their protocols on their systems. One possible reason for this may be that the developers of these protocols have not applied the formal methods that exist for verifying protocols. Furthermore, during this phase of the research, a lot of work has been done on the fairly new DNP-03 protocol, and many teething problems, therefore, complicated and deferred the final implementation of the outcome of this research.

In order for information to be delivered successfully between two devices, data streaming and data transactions are necessary. For reliable data streaming, one requires a synchronised state at both end-points. This synchronisation usually requires an open phase, a bi-directional data transfer phase, and a close phase. A transaction can be described as a sequence of two messages, one in each direction, interpreted as a request and a response. The open and close phases of a transaction are generally subsumed within the data transfer phase.

There are three main protocols currently used by the Electricity Network Management and Control (ENMAC) system in South Africa, of which the DNP-03 was above all introduced as an indirect demand by this research project, as well as the Integrated Remote Telecontrol Unit (IRTU) Recloser installations which have the ability to accommodate DNP-03. It also encourages enhancing future expansion and implementations of plant and RTU's. The protocols, as shown in figure 3.3 are:

- The Programmable Universal Telecontrol Unit (PUTU) Protocol.
- Eskom Telecontrol (ESTEL) Protocol.
- Distribution Network Protocol version 3 (DNP-03).
- MODBUS Protocol.

DNP-03 is the most recent protocol used to communicate with the RTU's. It was initially implemented during the course of this research, and forms part of the research results.

The fourth protocol (MODBUS) is used to interface to the ERTU Quad Integrated Communication Controller (QUICC) card, and was also just implemented during this research. This was also a ground-breaking experience, as excluding this development project, no other Intelligent Electronic Device (IED) has been tested or operational on any of the RTU's in the Eskom environment.

The MODBUS protocol is used to communicate from the IED to the RTU via a QUICC Controller card (Fig. 3.4), and was also pioneered and implemented during this research. The QUICC card comprises a microprocessor, memory, clock and mass storage device and is described in Chapter 4.13.3 [Ch.4, p. 116].

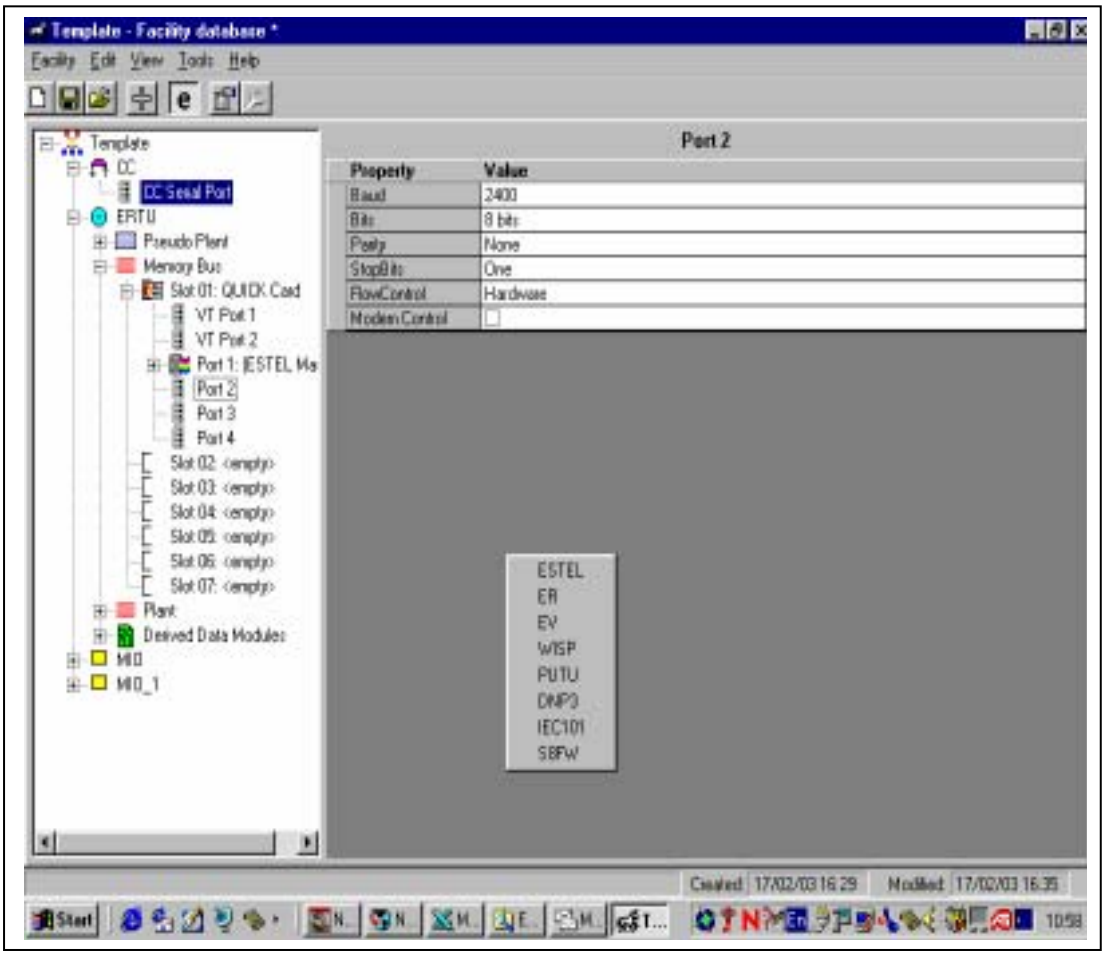


Fig. 3.3. The Protocol selection on a RTU using the UNICON program.

The protocol configurations comprise an integral element of the complete system development of this thesis, and therefore a brief explanation of each follows.

3.3.1. The Programmable Universal Telecontrol Unit (PUTU) protocol

A few major electricity, municipality and water suppliers in South Africa are successfully operating the PUTU telecontrol unit, dating from the late 1980's. PUTU RTU's are controlled via software on a MCT85 microprocessor controller card, with its main function being to process digital inputs, analogue inputs and relay outputs. The

Digital Input (DINP), High speed Scan (HSCAN) and microprocessor controller with Eprom 8085 (MCT85) cards achieve digital processing, and the DINP is scanned by the HSCAN, initiated by the MCT85 [34, p. 4/1].

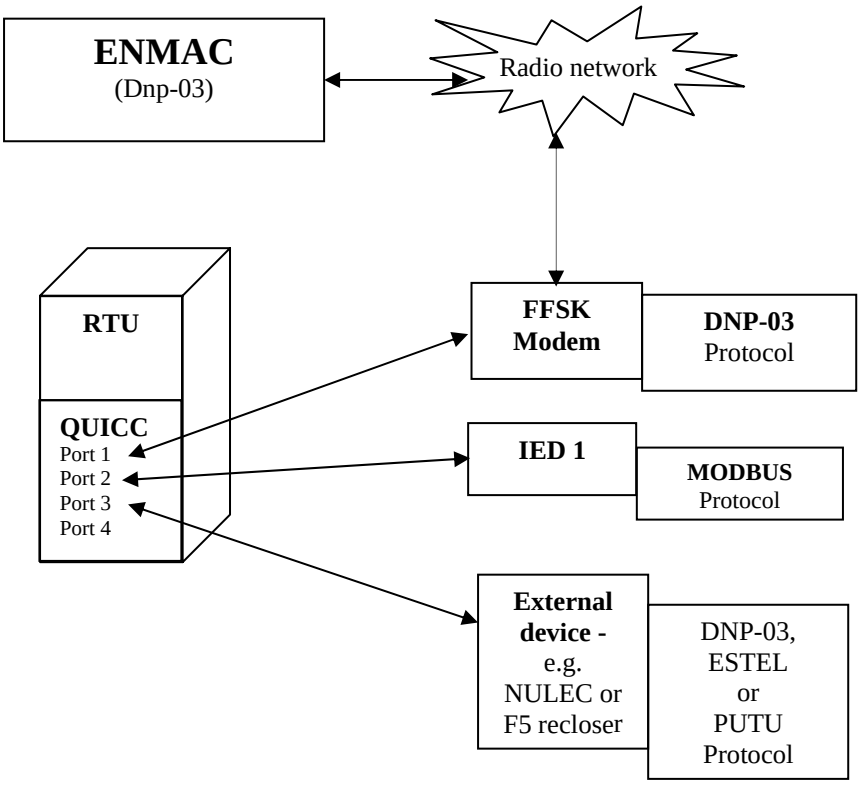


Fig. 3.4. Schematic layout of the protocol configuration as implemented for the IED.

The Master station initiates control sequences, and must issue the "proceed" command before the communications link is activated. To do any control functions, the ENMAC Front End Processor will send a "two-shot-control-pre-select" (TSCP) message to the RTU, which contains the control number.

The RTU responds with a "two-shot-control-pre-select acknowledge" (TSCPA) message upon successful execution. Communications are performed via Frequency Shift Keying (FSK) modems, at a selectable baud rate. This was proven to be a very

reliable protocol so far, and with this protocol, all messages have the following basic format [34, p. 15]:

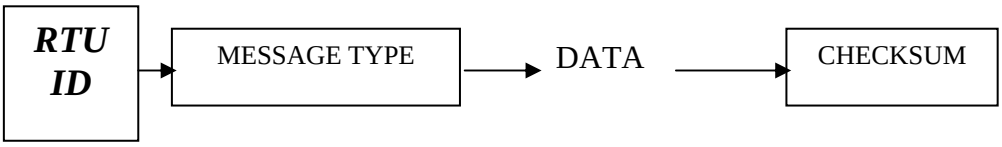


Fig. 3.5. PUTU protocol message format.

The first byte is the *RTU I.D*. The second byte is the *message type*, followed by the *data*. The last byte is an exclusive – or checksum [34, p. 16]. The Master station collects data from the RTU, using one of the following four General Poll (GP) messages [34, p. 16]:

- ***GP0 – Request priority 0 information only.***
- ***GP1 - Request priority 0 and 1 information.***
- ***GP2 - Request priority 0, 1 and 2 information.***
- ***GP3 - Request information, all priorities.***

In reply to a general poll (GP) by the ENMAC, all the following messages are sent by the RTU [34, p. 16]:

- ***Status Response Message (SRM)***
- ***Change Message (CM)***
- ***Database Synchronisation Message (DSM)***
- ***Update Message (UM)***
- ***Time Synchronisation Message (TSM)***

Plant controls are done by means of two-shot controls, in other words, two actions are necessary to execute a control command. This is to prevent accidental or unconfirmed execution of control commands. The PUTU protocol itself is still used effectively on many of the new generation RTU's.

3.3.2. Eskom Telecontrol (ESTEL) protocol

The ESTEL protocol was specifically developed for ESKOM, and is still used in a few RTU's. Most SCADA application functionalities include display control, data processing, real-time state estimation and automatic interpretation of alarms [8, p. 01]. For a telecontrol system to support its application functions it uses basic telecontrol functions which acquire data and forward commands by means of a protocol to remote terminal units. These functions deal with discrete state information, continuous variable quantities and integrated variables (digital, analogue and accumulator data) [8, p. 03].

With this protocol, the application layer has common layer services, which interacts with the basic telecontrol functions. These common services use basic layer functions such as queuing and polling to retrieve data from and execute commands at ESTEL Remote Terminal Units (ESTEL RTU's) [8, p. 04].

The most important common services provided by the application layer protocol to its user are as follows [8, p. 05]:

- ***Data Acquisition.***
- ***Supervisory Control.***

- ***RTU Configuration.***
- ***RTU Time Synchronisation.***
- ***RTU Status Monitoring.***
- ***Communications Network Control.***

These services make use of the basic layer services such as polling and queuing, and invoke the common services. The protocol procedures described here defines a selection of basic application functions that utilise standard communication services like:

a) RTU Initialisation Sequence

Once the Master receives notification that the RTU has powered up, it will attempt to download the RTU with the relevant processing parameters, and may also attempt to test the initialisation status of the RTU by requesting the RTU to return its stage number with associated database reference number [8, p. 06]. This was found to normally occur after a communications time out. If the Master detects a difference, i.e. databases have changed since the last communication then the RTU can be synchronised from start by sending a DEFAULT message. The Master can also complete the synchronisation from the last stage number [8, p. 06].

If communication is lost for a short period of time during the loading of a RTU, and is re-established without the RTU going through its power-up sequence; then:

- ☞ The Master will ask the RTU for its current load stage number by sending a STAGE message with the stage field set to zero [8, p. 07].
- ☞ Loading of the RTU then continues from the stage reported unless the returned database revision numbers differs from the Master's [8, p. 08].
- ☞ The Master will send a DEFAULT message to reset the RTU's configuration. Initialisation is then started from stage 0 [8, p. 08].

b) <u>Queuing and queue servicing procedures</u>

There are two queues for data per communication line [8, p. 14]:

- 🕮 Low priority messages are queued in the synchronous buffer.
- 🕮 High priority messages are queued in the pre-emptive buffer.

The Master station services low-priority data in the synchronous queues for each link in a regular fashion. When it is activated to service the next set of queues of a link it will scan the queues for that link to find any data destined for that RTU [8, p. 15].

c) <u>Polling cycle and types of polls</u>

The predetermined sequence of issuing polls is known as the ***polling cycle***, which exists per communication line. The Master services the polling cycle, i.e. issues polls, either regularly or as replies to previously issued polls are received [8, p. 14]. When activated to poll a RTU, the Master determines if that RTU has replied to the previous poll. If this is the case it issues either a synchronous or pre-emptive poll which is

determined by the setting of the pre-emptive flag. This flag, if set, changes status based on either:

- A time out to receive the required reply to a pre-emptive request, or
- When the required request is received [8, p. 14].

3.4. Distribution Network protocol (DNP-03)

GE Harris (the former Westronic, Inc.), now GE, originally created the Distribution Network Protocol (DNP). The "DNP-03 Basic 4" protocol specification document set was released into the public domain, and a newly formed DNP-03 users group was given ownership of the protocol [41, p. 1]. Since then, the protocol has gained international acceptance [41, p. 1]. DNP-03 is an open telecontrol protocol that is extensively used and in general considered to be the standard international telecontrol protocol. In South Africa, Eskom Distribution decided during 1999 to adopt the DNP-03 protocol. This followed the adoption of DNP-03 by the NRS. Implementation of DNP-03 in the South African environment has however been loaded with problems; the most significant of which has been supplier non-delivery.

One of the main drivers behind the adoption of an open standard protocol was the advent of Auto Recloser Controllers with integrated telecontrol capability. Such a device saves the cost of a separate pole mounted RTU (if on telecontrol). Many other benefits also flow from the use of open protocols such as the freedom to choose equipment from many suppliers. DNP-03 is an open, robust, intelligent, and efficient modern SCADA protocol, with the following features:

- It can request and respond with multiple data types in single messages [41, p. 2].
- It can segment messages into multiple frames to ensure excellent error detection and recovery [41, p. 2].
- It can include only changed data in response messages, assign priorities to data items and request data items periodically based on their priority [41, p. 3].
- It can respond without request (unsolicited), support time synchronisation and a standard time format [41, p. 3].
- It can also allow multiple Masters and peer-to-peer operations, and allow user definable objects, including file transfer [41, p. 2].

Instead of adhering to the Open System Interconnection (OSI) 7-layer protocol, DNP-03 remains with a straightforward 3-layer standard [41, p. 2], with a fourth layer, a pseudo-transport layer that allows for message segmentation [7, p. 10].

APPLICATION LAYER
PSEUDO-TRANSPORT LAYER
DATA LINK LAYER
PHYSICAL LAYER

Fig. 3.6. The Enhanced Performance Architecture (EPA) model of DNP-03.

In figure 3.6, one can describe the Enhanced Performance Architecture (EPA) DNP03 model, starting from the bottom layer (Physical) to the top (Application) layer:

- The **physical layer** is primarily concerned with the physical media over which the protocol is being communicated. For example, it handles the state of the media (clear or busy) and the synchronisation across the media (starting and stopping)

[41, p. 2]. Most commonly, DNP is specified over a simple serial physical layer such as RS-232, also using physical media such as fiber or radio [41, p. 1].

🕮 The **data link layer** manages the *logical* link between sender and receiver of information [41, p. 3], and it improves the physical channel error characteristics [7, p. 11]. This is accomplished by beginning each data link frame with a data link header. A frame is a portion of a complete message communicated over the physical layer, and the maximum size of a data link frame is 256 bytes [7, p. 11]. Each frame has a 16-bit source address and a 16-bit destination address, which may be a broadcast address [41, p. 3], with the following features:

- ⊠ Manages the link between devices.
- ⊠ Deals with addressing.
- ⊠ Start characters (0 x 0564).
- ⊠ Manages the message length.
- ⊠ Contains the Control byte. The Control byte in turn:
  - Indicates the type of frame.
  - Show the direction of the frame.
  - Shows the flow control information.

Start (0x05)
Start (0x64)
Length
Control
Destination LSB
Destination MSB
Source LSB
Source MSB
Cyclic Redundancy Check (CRC)
User Data Blocks
Cyclic Redundancy Check (CRC)

Fig. 3.7. The DNP03 Data link layer structure [42, p. 4].

- The **pseudo-transport layer** indicates the total length of received messages because it appends data link layer frames - each with their own indicated length [42, p. 4]. It also allows for message fragmentation, and indicates if the frame is first or last. This layer also increments the sequence number to manage message fragmentation [42, p. 4].

- The **application layer** responds to complete messages received (and passed up from the transport layer) [41, p. 4], and builds messages based on the need for or the availability of user data [7, p. 26]. Once messages have been built, they are passed down to the pseudo-transport layer where they are segmented and passed to the data link layer and eventually communicated over the physical layer [41, p. 4]. When the data to be transmitted is too large for a single application layer message, multiple application layer messages may be built and transmitted sequentially [7, p. 21]. However, each message is an independent application layer message; their only association with each other is an indication in all but the last message that more messages follow [41, p. 4]. Because of this possible fragmentation of application data, each application layer message is referred to as a fragment and a message may either be a single-fragment message or a multi-fragment message [41, p. 8]. Application layer fragments from Master DNP-03 stations are typically requests for operations on data objects, and application layer fragments from RTU DNP-03 stations are typically responses to those requests [41, p. 4]. A RTU DNP-03 station may also transmit a message without a request [7, p. 49]. An **unsolicited** response is described in chapter 3 [Ch.3, p. 57].

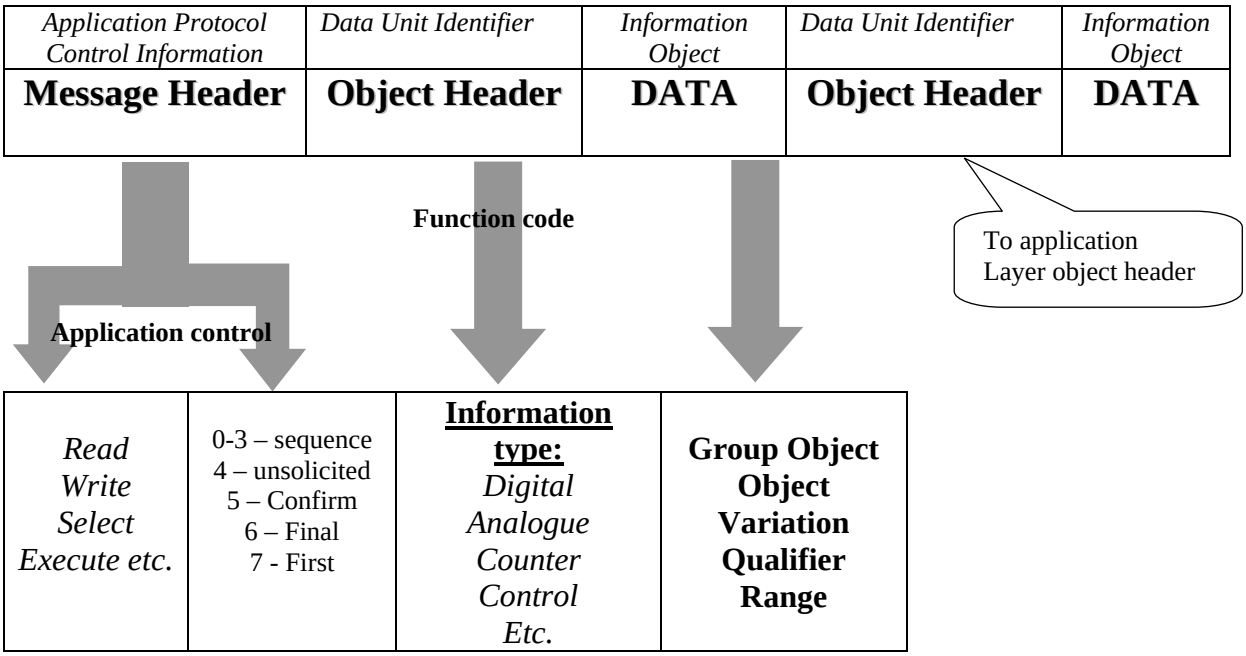


Fig. 3.8. Diagrammatical explanation of the Application layer message header structure [42, pp. 6-8].

While DNP-03 allows multiple data types in a single message, it only allows a single requested operation on the data types within the message, with the following function codes:

- Confirm (for application layer confirmations) [7, p. 32].
- Read and write, select and operate (for select-before-operate (SBO), controls) [7, p. 32].
- Direct operate (for operation of controls without SBO) [7, p. 33].
- Enable and disable unsolicited messages, and assign class discussed next [7, p. 46].

The application layer header function code applies to all object headers, and therefore all data within the message fragment [41, p. 5]. In DNP-03, data is organised into data types [7, p. 5]. Each data type is an object group, including [41, pp. 60-61; 70-73]:

- Binary inputs (single-bit read-only values).
- Binary outputs (single-bit values whose status may be read, or that may be pulsed or latched directly).
- Analogue inputs (multiple-bit read-only values).
- Analogue outputs (multiple-bit values whose status may be read).
- Counters.
- Time and date.
- File transfer objects.

For each object group, or data type, one or more data points exist. A data point is a single data value of the type specified by its object group [41, p. 5]. Also within each object group, object group variations exist [7, p. 67], like variations of analogue inputs, allowing for transfer of the data as 16-bit signed integer values, 32-bit signed integer values, or as 32-bit floating point values [7, p. 67].

Some application layer header function codes indicate that object data follows each object header; other function codes indicate that there is no object data in the message – instead multiple object headers, if present, follow each other contiguously [41, p. 5]. For instance, a read request message fragment only contains object headers that describe the object

groups, variations, and point ranges that are requested to be read and responded to. A read response message fragment contains object headers and the request object data [7, p. 66].

3.4.1. DNP-03 object ranges

The DNP-03 protocol allows object point ranges to be specified in a variety of ways [41, p. 5]:

For request messages: Object point's ranges may consist of a request for different points of the specified object group [7, pp. 66-67]:

- A request for a contiguous range of points (beginning with a specified starting point, and ending with a specified stopping point).
- A request for a maximum quantity of points, or with a list of requested points.

For response messages: Object point ranges typically consist of either a contiguous range of points beginning with a specified starting point and ending with a specified stopping point, or with a list of points.

- For response object point ranges that consist of a list of points, a point number precedes each data object [7, p. 66].
- The number of points in the list is specified as part of the object point range.

For each change data point, a time can be associated with the change; detection of a data value that changes is considered a change event [41, p. 5]. At any given time, it is possible to have multiple change events for some points, and no change events for other points [41, p. 5]. In DNP-03, object groups, and the data points within them, can be

further organised into classes [7, p. 46]. This provides an efficient method of requesting data; a simple message can be sent to request all data in a specific class, and can be referred to as scanning for class data.

3.4.2. Classes defined in DNP-03

There are four classes defined in DNP-03 [41, p. 6]:

- Class 0 represents all static (not change event) data.
- Classes 1, 2, and 3 all represent different priorities of change event data.

By associating different change event data with different classes, the classes can be requested with varying periodic rates. Assuming that class 1 contains the highest priority change event-data, and class 3 contains the lowest priority change event-data, then:

- A class 1 poll would ideally be performed as often as possible [41, p. 6].
- A class 2 poll would be performed less often, and a class 3 poll would be performed even less often [41, p. 6].
- For each class data response, only the class data that has changed will be returned (keeping the response messages small and efficient) [41, p. 6].

Conclusively, to acquire data not associated with either class 1, 2, or 3, an integrity poll, consisting of a class 0 scan would be performed [7, p. 26], because of a possible large amount of data that can be returned [41, p. 6]. However the bandwidths available to

most utility companies are still quite limited and DNP-03 provides several different means of retrieving data [41, pp. 57-59]:

1. ***Quiescent Operation***, in which the Master never polls any Slave, and all communication is unsolicited report-by-exception. The Master still sends application layer confirmations to the Slave [41, p. 58].
2. ***Unsolicited Report-by-Exception Operation***, in which most communication is unsolicited, but the Master occasionally sends integrity polls for Class 0 data to verify its database is up to date [41, p. 58].
3. ***Polled Report-by-Exception Operation***, in which the Master polls frequently for event data and occasionally for Class 0 data. Most often the Slave's response to the event polls will contain few objects, so polling can be very quick [41, p. 58].
4. ***Polled Static Operation***, in which the Master polls only for Class 0 data or the specific data it requires. This method is the simplest to implement, but can be very inefficient if the number of points that need to be retrieved is high and changes are infrequent [41, p. 59].

Some systems may also choose to mix Polled and Unsolicited Report-by-Exception operation. Given that unsolicited responses are not always possible using some physical layers, the recommended minimum implementation for a Slave is polled Report-by-Exception operation. According to the DNP-03 users group, the subset definitions have been written with this in mind [41, p. 58].

3.5. The MODBUS protocol

This section considers the MODBUS serial RS232 and RS422/485 as used by this development. MODBUS is a simple networking protocol that allows for control and transfer of data between MODBUS devices. Enhancements to MODBUS include MODBUS Plus and MODBUS /TCP protocols. MODBUS Plus communicates via a single twisted pair of wires and uses a token passing sequence for peer communication sequences. MODBUS /TCP are an open standard designed to facilitate MODBUS message transfer using TCP/IP protocol and standard Ethernet networks [23, p. 4-7]. It is not viable to deal with this protocol in detail for the purpose of this script; therefore only informative detail will be presented to familiarise the reader with the most significant features.

The MODBUS protocol is used as the interface between the IED development and the RTU QUICC [Ch. 4, p. 120], and defines a message structure that controllers use, regardless of the type of network available. It describes the process a controller uses to request access to another device, and establishes a common format for the layout and contents of message fields [23, p. 5].

3.5.1. MODBUS serial RS485 and RS232 communications

Electronic data communication between elements generally falls into two broad categories [23, p. 1]:

- Single-ended (RS232 and RS485) and
- Differential (RS422).

The RS232 signals are represented by voltage levels with respect to a system common (power / logic ground). The "idle" state (MARK) has the signal level negative with respect to common, and the "active" state (SPACE) has the signal level positive with respect to common. It has numerous handshaking lines (primarily used with modems), and also specifies a communications protocol [23, p. 1].

RS485 meets the requirements for a truly multi-point communications network, and the standard specifies up to 32 drivers and 32 receivers on a single (2-wire) bus. RS485 extends the common mode range for both drivers and receivers in the "tri-state" mode and with power off. Also, RS485 drivers are able to withstand "data collisions" (bus contention) problems and bus fault conditions [39, p. 3]. RS485 is a balanced line, half-duplex transmission system allowing transmission distances of up to 1.2 km [23, p. 1].

In practice RS485 and RS422 are very similar to each other. The main working difference is that RS485 is used for 2-wire multi-drop half duplex systems and RS422 is used for 4-wire point to point full duplex systems [23, p. 9].

3.5.2. The MODBUS message

A Master initiates communication on a Modbus Network with a "query" to a Slave. The Slave, which is constantly monitoring the network for “Queries”, will recognise only the "Queries" addressed to it, and will respond either by performing an action or by returning a "response". Only the Master can initiate a query (poll or instruction) [23, p. 2], which contains the following [23, p. 5]:

- The device address.
- A function code defining the requested action.
- Any data to be sent.
- An error-checking field.

The response in return contains [23, p. 5]:

- Fields confirming the action taken.
- Any data to be returned.
- An error-checking field.

If an error occurred in receipt of the message, or if the Slave is unable to perform the requested action, it will construct an error message and send it as its response [23, p. 5]. A representation of a Master to Slave query is shown in figure 3.9, while a representation of the Slave response to the Master shown in figure 3.10.

Slave Address	Function Code	Start Address (Hi)	Start Address (Lo)	Number of Points (Hi)	Number of Points (Lo)	Error Check (Lo)	Error Check (Hi)

Fig. 3.9. Format of a MODBUS Master to Slave (remote unit) query [23, p. 6].

The Master to Slave query message shown in figure 3.9 can be explained as follows [23, p. 6]:

- **Slave Address:** 8-bit value representing the Slave being addressed (1 to 247), 0 is reserved for the broadcast address.
- **Function Code:** 8-bit value telling the addressed Slave what action is to be performed.
- **Start Address (Hi):** The top (most significant) eight bits of a sixteen-bit number specifying the start address of the data being requested.
- **Start Address (Lo):** The bottom (least significant) eight bits of a sixteen-bit number specifying the start address of the data being requested.
- **Number of Points (Hi):** The top (most significant) eight bits of a sixteen-bit number specifying the number of registers being requested.
- **Number of Points (Lo):** The bottom (least significant) eight bits of a sixteen-bit number specifying the number of registers being requested.
- **Error Check (Lo):** The bottom (least significant) eight bits of a sixteen-bit number representing the error check value.
- **Error Check (Hi):** The top (most significant) eight bits of a sixteen-bit number representing the error check value.

Slave Address	Function Code	Byte Count	Data (Hi)	Data (Lo)	Error Check (Lo)	Error Check (Hi)

Fig. 3.10. Format of a MODBUS Slave (remote unit) response to the Master.

The Slave response message to the Master as shown in figure 3.10 can be explained as follows [23, p. 6]:

- **Slave Address:** 8-bit value representing the address of Slave that has just responded.

- **Function Code:** 8-bit value which, when a copy of the function code in the query, indicates that the Slave recognised the query and has responded.
- **Byte Count:** 8-bit value indicating the number of data bytes contained within this response.
- **Data (Hi):** The top (most significant) eight bits of a sixteen-bit number representing the register(s) requested in the query.
- **Data (Lo):** The bottom (least significant) eight bits of a sixteen-bit number representing the register(s) requested in the query.
- **Error Check (Lo):** The bottom (least significant) eight bits of a sixteen-bit number representing the error check value.
- **Error Check (Hi):** The top (most significant) eight bits of a sixteen-bit number representing the error check value.

There are two MODBUS serial transmission modes, ASCII and RTU, of which the RTU mode is usually the preferred mode, with a shorter message length than ASCII. ASCII serial transmission mode was used due to the Video Terminal program capturing criteria and will be discussed briefly.

3.5.3. ASCII MODBUS serial transmission modes

In the ASCII Transmission Mode (American Standard Code for Information Interchange), each character byte in a message is sent as two ASCII characters [23, p. 7]. When messages are transmitted on standard MODBUS serial networks, each character or byte is sent in the order (left to right) as shown in figure 3.11.

Least Significant Bit (LSB),								*Most Significant Bit (MSB).*		
a) Start	1	2	3	4	5	6	7	8	Par	Stop
b) Start	1	2	3	4	5	6	7	8	Stop	Stop
c) Start	1	2	3	4	5	6	7	8	Stop	

Fig. 3.11. Serially transmitted characters [23, p. 9]:
a) With Parity Checking
b) Without Parity Checking, 2 Stop Bits
c) Without Parity Checking, 1 Stop Bit.

3.5.4. Function codes

The function code part of a MODBUS message defines the action to be taken by the Slave, and support the following function codes as shown in table 3.2:

Table 3.2. Function codes of the MODBUS messages [23, p.11].

Code (HEX)	MODBUS name	Description
03	Read Holding Registers	Read the contents of read/write location (4X references)
04	Read Input Registers	Read the contents of read only location (3X references)
16	Pre-set Multiple Registers	Set the contents of read/write location (4X references)

3.6. The Electricity Network Management and Control (ENMAC) system

It is required that the reader be introduced to the existing SCADA system with which this development was integrated before considering the development done on the IED.

The Electricity Network Management and Control (ENMAC) SCADA Master system in the North Western Region utilises two control centres, which are interconnected via a high-speed data link, backed up by an Integrated Services Digital Network (ISDN) link. On this system,

various brands of RTU's were in operation at the start of this project, e.g.: INTRAC, PUTU, MRTU, ERTU and DRTU. All of these are communicating to the ENMAC Master system via different protocols as explained earlier in this chapter. These RTU's operate on different communications media like fiber optic, microwave or radio communications.

The RTU's monitor the analogue and digital changes at distant locations, and have the facility to do control functions demanded by the control centre. Voltage Transformers (VT's) and Current Transformers (CT's) are used to facilitate connection of instrumentation and protection circuitry to the high voltage conductors. This is usually arranged to give full-scale deflection at 110V or 1A respectively. This is then converted by a Voltage Transducer to a 5mA rating. MW, MVAr, Amp and Voltage transducers are equipped between the CT and VT, to interface with the instrumentation. The latter is connected to the RTU for visualising at the ENMAC Operator Workstation (OW) at the Control centre. Figure 3.12 is the display of a substation on the ENMAC operator workstation (OW), and shows the typical visual layout of such measurements. Digital inputs on the RTU's can be time-tagged, which means that the Master station and the RTU are synchronised with each other. This allows that the actual time of an event can be obtained per digital input with an accuracy of 2-5 milliseconds.

Whilst doing some communications tests during this research project, it was found that if two different protocols, e.g. DNP-03 and INTRAC were used on the same channel, data-error problems occurred, and it was decided to phase out all the old INTRAC and PUTU RTU's. To aggravate this, the course of the research was altered leading to drastic modifications and scope changes on the existing development to date. To have a contemporary and functional measuring unit, the changes necessitated that the end result

must now incorporate DNP-03 RTU's, without singular digital hardware interfaces but using IED ports instead.

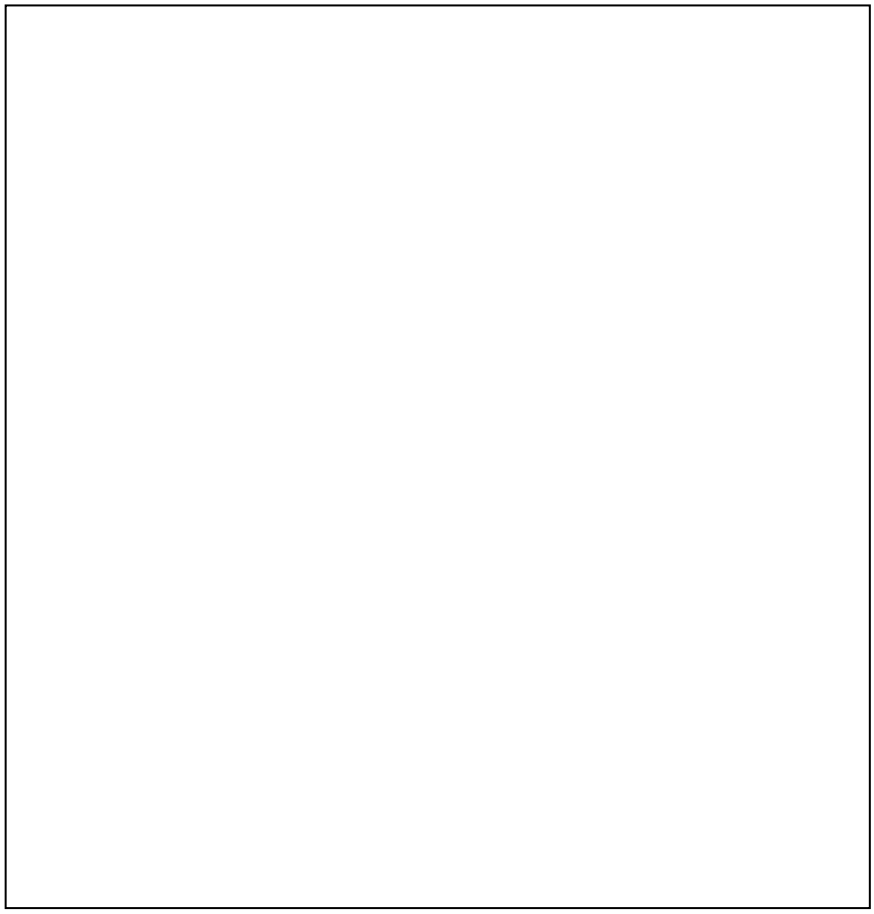


Fig. 3.12. Magnified view of analogues and indications on the ENMAC.

On a typical RTU, the voltage measurements are also carried out by means of an ammeter [36, p. 89], and capacitance voltage-to-current converters are used for AC measurements only, while resistance voltage-to-current converters are used for DC measurements [36, pp. 89-91]. Voltage-to-current conversion is used effectively to reduce losses due to long wire connections at all the major substations equipped with SCADA [24, p.12].

At a first glance, it seems a very straightforward solution to measure the voltage dips and surges with the normal analogue values on the existing SCADA system. This option was

strongly considered, but due to the limited amount of analogue inputs per RTU (128 maximum), as well as the fact that it does not allow for the capability of harmonic measurements, a different alternative was searched for.

3.7. Summary

In this chapter the reader was introduced to different SCADA Master to RTU protocols, as used for the communication of the developed QOS Measuring unit. This incorporates MODBUS, ESTEL and DNP-03. DNP-03 is a relatively high-overhead protocol, compared to its predecessors and used for the RTU Remote / Master station communications.

DNP-03 provides several different means of retrieving data, like quiet operation, where the Master polls the RTU only at a preset time (e.g. once every 12 hours), and the RTU communications is unsolicited report-by-exception. The RTU will only report if an event occurred at the station.

In Solicited mode, the Master polls frequently for event data. During Polled Static operation, where the Master polls only for Class 0 data, it requests only for the specific data it requires. Each RTU receives a poll from the Master station before data is transmitted. In the case of unsolicited report by exception, the RTU responses on any changes from the field.

The role of the NRS was also discussed in this chapter, explaining the NRS-048 requirement for improved harmonic and voltage dip limits. As the probability of

harmonic problems is realistically high in most power systems, the instances in which they occur may result in power quality problems. These limits can reduce power system reliability and efficiency. According to the NER guidelines, the use of DSP techniques can assist in measuring the quality of supply as well as envision the information on the ENMAC system.

Although interruptions and outages can be monitored and confirmed, it is imperative to emphasise that only through customer co-operation and involvement, can these solutions be successfully implemented for the prevention of outages, plant damage and production loss.

CHAPTER 4

METHODS & TECHNIQUES

4.1. Introduction

The techniques that were used for the Intelligent Electronic Device (IED) development, as well as for implementing it on the SCADA system, are described in this chapter. The initial objective of the research, though, was to implement a trouble-free, robust and low-cost hardware solution. Unfortunately, due to technology and protocol changes during the time of this research, the final outcome was based on a combination of both hardware and software techniques, complicating the primary proposal.

It was also during this time that the author decided not to go above the 13^{th} harmonic for the harmonic measurements, because the author only wanted to prove that the software was suitable for the IED, and to illustrate that the concept is a workable solution for measuring the quality of electricity supply via SCADA. This decision was substantiated when some historic trends in the North Western Region were examined, and the author came to a conclusion that this would actually not undermine the project outcome.

Another significant motive was that the author was expected to cut the development costs to a minimum.

4.2. Sequence and approach of the development board

Figure 4.1 shows the sequence and the approach in which this development was executed, starting with the input circuitry of the measurement unit to the Substation plant. The developed input circuitry was integrated to the measurement circuitry; initially the ADSP-2181 EZ-KIT Lite and software, to measure the station signal (110VAC) required to be tested. Due to difficulty in proving the measurement accuracy and reliability, a test program was then developed by the author, using LINUX to generate waves to simulate real harmonics and voltage dip circumstances on a PC. Upon interfacing the ADSP-2181 EZ-KIT Lite to the RTU, the interface port unfortunately got damaged, and the ADSP-2181 EZ-KIT Lite measurement circuitry was then discarded.

Nevertheless, the same input circuitry was still used, and the author assembled new measurement circuitry to replace the ADSP-2181 EZ-KIT Lite. The Axiom CME12-A4 development unit was chosen, using a single phase only for the initial tests, to test with the developed LINUX test program. After satisfactory laboratory tests and demonstration of accuracy was obtained on the Axiom CME12-A4 single phase development breadboard, the author then integrated this to the RTU. The combination of the developed input circuitry with the prototype single phase CME12-A4 measurement unit tested adequately according to the NRS specifications and the concept was then used for the final three phase product, the IED.

The prototype (single phase bread boarded version) shown in figure 4.5 proved to comply with the criteria laid down by the NRS, as shown in the results [Ch. 6.2.2, p.181], and the final 3 phase product, as shown in figure 4.6 was built and integrated to the RTU. This was then configured and tested to the ENMAC Master station.

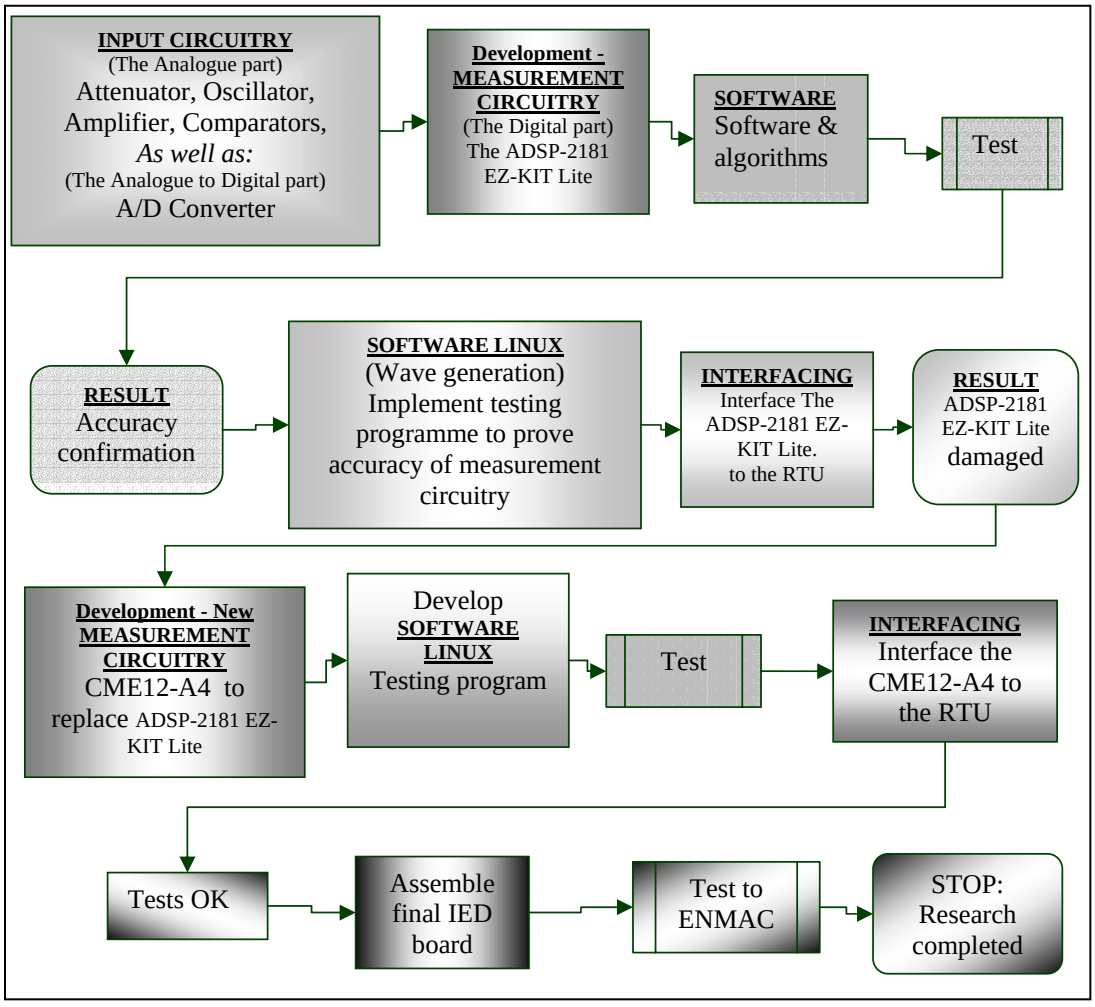


Fig. 4.1. Diagram showing the sequence and the approach in which the IED was developed.

4.3. Layout of this chapter

The outlay of this chapter can be divided into 3 parts, i.e. the hardware development, the simulation software development and the software configuration development and configurations. The content is shown in figure 4.2.

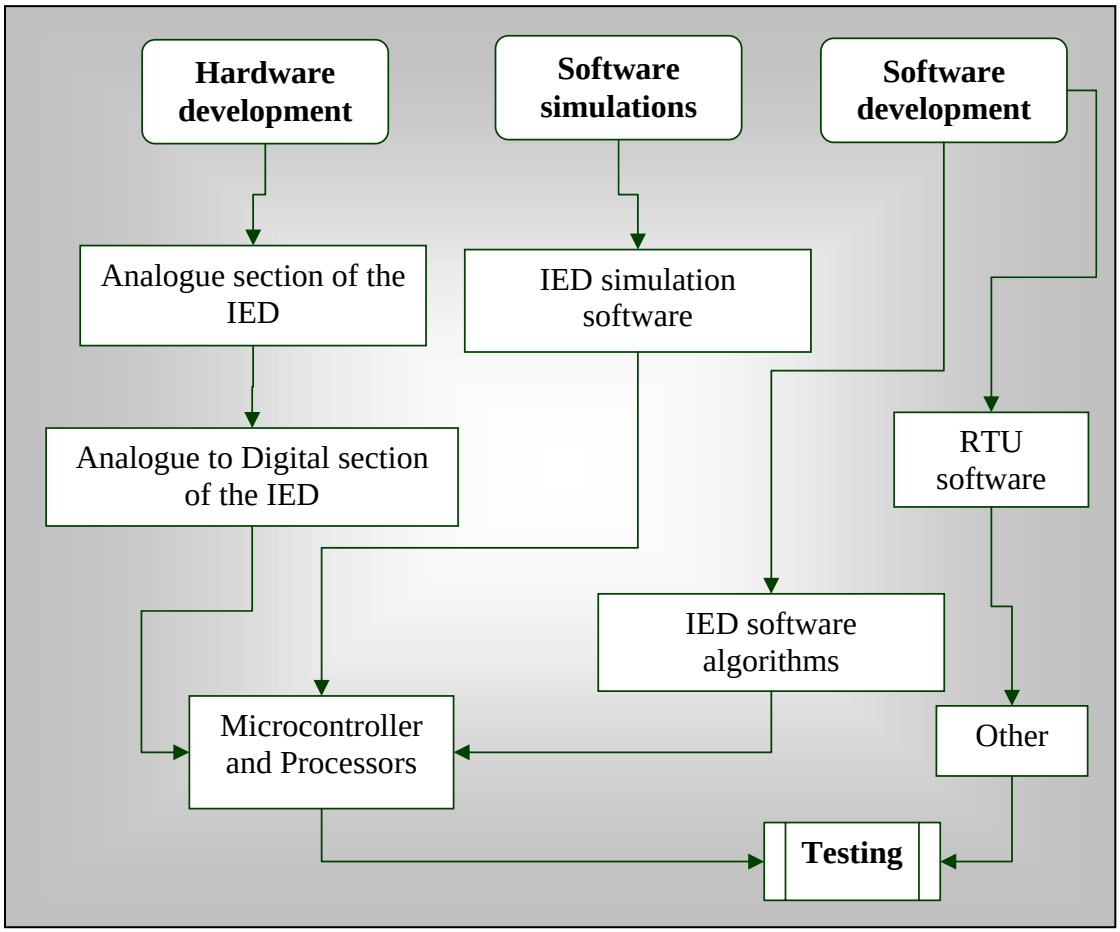


Fig. 4.2. Block diagram, showing the layout of this chapter, indicating the integration of each section until the final testing stage on the IED.

During this chapter flowcharts are used to visually represent the flow of data and algorithms through the design phase of the programming and operations performed, and the sequence in which they are performed.

4.4. IED hardware development

Apparently what was needed for this development was a device that would be able to:

- Attenuate the 110VAC signals from the 110VAC of the substation VT.
- These signals were then filtered to enable proper measurements as specified by the NRS criteria.
- The filtered signals were then converted from analogue to digital with an A/D converter.
- The A/D results had to be further processed with a Microcontroller.
- This must then be presented to the RTU, to enable the ENMAC visualisation of these readings.

On the hardware phase of the research project, it was obvious that the requirement of this project was a classic application for Digital Signal Processing (DSP), and the aim was to use the most inexpensive DSP capable of performing the task at hand.

4.4.1. The analogue section

Figure 4.3, given in the original NRS specification, define the required dip performance. A 50Hz wave has a period of 20ms. In terms of the NRS specifications [Ch. 2, p.38] a dropout of 1 wave cycle is acceptable, and the sampling window of 3000ms equates to 15 complete wave cycles.

These requirements suggested that one needed to implement a low-pass digital filter operating in the range of 3 – 50 Hz to detect the fluctuations.

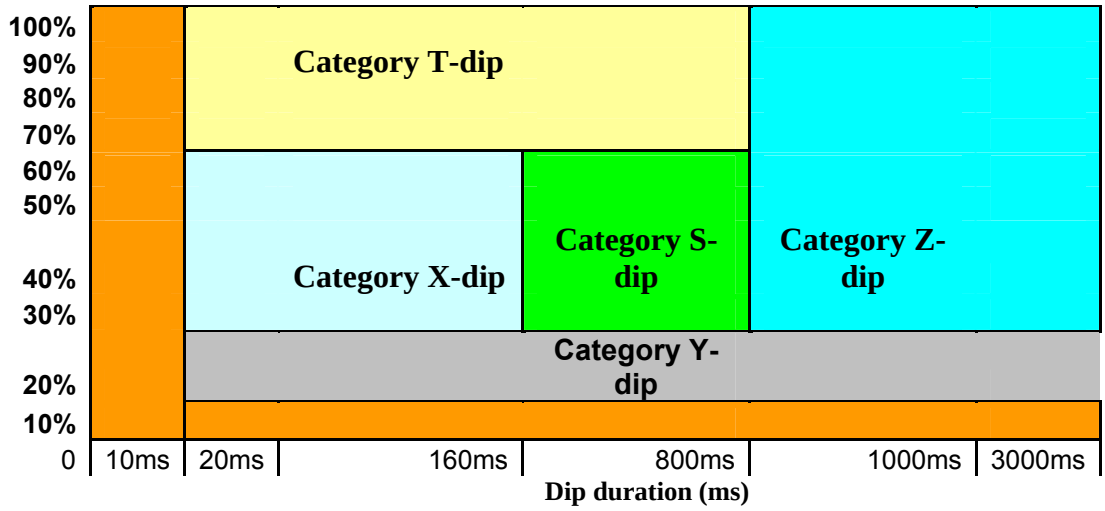


Fig. 4.3. NRS specifications for measuring the voltage dip performance.

For the development of the filter circuitry, the two options that could comply with the requirements needed to implement a low-pass digital filter, operating in the range of 3 – 50 Hz to detect the fluctuations, were considered:

- Windowed -Sinc filter.
- Low Pass Butterworth filter (LTC1063).

4.4.1.1. Windowed Sinc filter

According to Smith [38, p. 286], the Windowed-Sinc filters are used to separate one band of frequencies from another, and are very stable. The author therefore investigated the option of using a Windowed-Sinc filter, as it is normally used in

this type of application. After detailed analysis on this option was done, the author finally decided against it for three reasons:

- ✔ The filter consumed more processing power than the author would have ideally liked to be allocated for the purpose.
- ✔ The author felt that simpler algorithms could perform just as well.
- ✔ The dip envelopes were not always necessarily sinusoidal, and if they occurred quickly, the low-pass filter would probably smooth their sharp edges.

Option 2, the low-pass Butterworth filter (LTC1063) was to be implemented.

4.4.1.2. The 5th order low-pass Butterworth filter

The filter used was a LTC1063 from Linear Technology, and the pin layout is shown in figure 4.4. It is clock tuneable, has 12-bit accuracy, low noise, and low dc offset. It was selected according to the following criteria:

- ☞ Provide minimal attenuation of the 13th harmonic at 650Hz.
- ☞ Provide maximum attenuation of frequencies that will be aliased back onto the used frequencies - say 1% or -40dB.
- ☞ At 64 samples per cycle the sampling rate is 3200Hz and the aliasing frequency is 1600Hz.

The LTC 1063 was used in conjunction with a program called FilterCad-03, which is a computer aided design program to help users in filter design to create filters using Linear Technology's monolithic filter ICs. This program helped a great deal to achieve better results by providing the ability to perform "what if" scenarios with the configuration and values of various components. By using this program, choosing a -3dB cut-off frequency of 900Hz on the LTC 1063, resulted in the following:

- The loss at the 13th harmonic at 650Hz was <0.2dB (1%) and negligible below 500Hz.
- 2550Hz aliased onto the 13th harmonic at 650Hz at a value of –45 dB (0.6%).
- 3150Hz aliased onto the fundamental at -55dB (0.2%).

The 12-bit digitisation and integer calculations have the same order of magnitude error. On the development prototype layout, the analogue section is the input from the substation VT, and connects to the VT 110VAC mains via suitable input isolation. The circuitry has an input buffer amplifier which can be adjusted to accommodate a wide range of input isolation devices, in the range +-1Volt to +-20Volts.

The inclusion of a zero-crossing comparator is shown in figure 4.4. This allows the IED to "lock on" to the actual frequency of the mains and therefore eliminates drift error. The LTC1063 low-pass filter as described follows the

buffer, in order to limit the high frequency content of the signal to the analogue to digital converter (ADC).

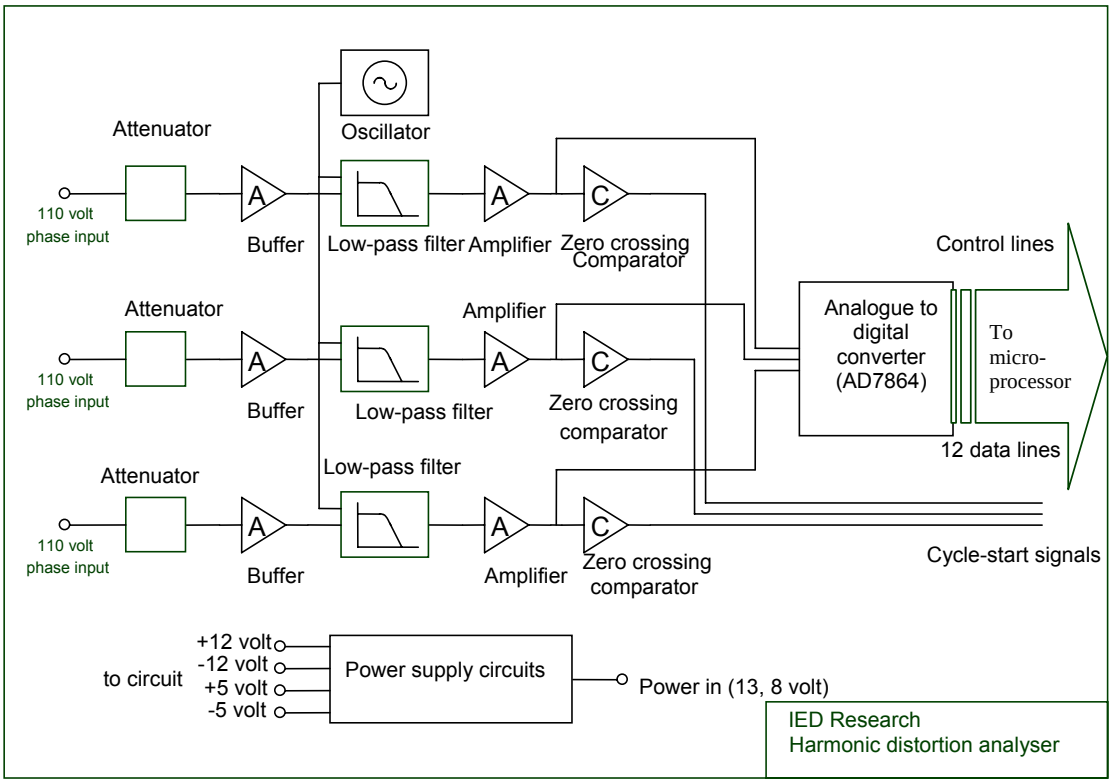


Fig. 4.4. Three phase hardware layout of the input circuit analogue section, as well as the analogue to digital section.

On the hardware layout shown in figure 4.4, the prototype unit (Axiom CME12-A4) input circuitry, as described in the analogue section, was built on a prototyping board, and only one of the three phases assembled. On the final development, shown in figure 4.6, all three phases were incorporated in the IED while using the MC68HC812A4CPV8 Microcontroller. The ADC (AD7864AS) used, has 12-bit resolution, and samples the three phases simultaneously. A zero-crossing detector provides a synchronisation signal for the start of every cycle, and peripheral components for the filter clock and power supply generation.

4.5. The analogue to digital section

The Analog Devices AD7864 is a 12-bit 4 channel simultaneous sampling device of suitable specification. The converter performs an A-D conversion in about 6 µsec's. This device has built-in voltage reference; and the configuration is pin programmable, and is very simple to interface with only 3 control lines. Some features of the AD7864AS ADC include the following:

- High speed (1.65 µs) 12-Bit ADC
- 4 Simultaneously sampled snputs
- 4 Track/Hold amplifiers
- HW/SW Select of Channel sequence for conversion.
- Single-supply operation.
- Low Power, 90 mW.
- Over voltage protection on the analogue Inputs.

4.6. Processors and microcontrollers

The author initially decided to first implement the most CPU-intensive requirement, the harmonic content, in this project, in order to establish the overall viability. Originally the ADSP-2100 family of DSP processors were chosen for the project, due to availability, cost and local support. Problems on the initial ADSP-2181 EZ-KIT Lite board kit eventually influenced us to consider other options, resulting in the employment of a second development board. The two options of development boards that were tested prior to the final construction of the IED board were:

- The ADSP-2181 EZ-KIT Lite.
- The CME12-A4 by Axiom Manufacturing.

4.6.1. ADSP-2181 EZ-KIT Lite

ADSP-2100 Family DSP's are well suited for applications that detect sinusoidal tones. Digital tone detection applications usually have fast execution speeds and require minimum memory storage. One can take advantage of these features by coding tone detection as a sub task of a larger, single-chip DSP application, or by using a single DSP to simultaneously handle tone processing for many independent channels. All program memory and data memory requirements are easily fulfilled by the on-chip memory of the ADSP-2100 Family processors, leaving the remaining on-chip memory space for other DSP functions.

Considering these, the ADSP-2181 EZ-KIT Lite evaluation system, using the ASDP-2100 Family processors, was implemented as a rational option for a 16-bit fixed point DSP microcomputer. The evaluation board was designed to be used in conjunction with VisualDSP® and the 16-bit tools as a complete code evaluation and debug system. Using the EZ-KIT Lite with the debugger, the ADSP-2181 were observed to execute programs from on-chip RAM, interacting with on-board devices, and communicating with other peripherals located on optional add-on modules [2, p. 4].

The PC has easy access to the ADSP-2181 processor through a serial port. The monitor program enables complete target debug capability through the serial port.

In contrast, the emulator allows the PC to perform in-circuit emulation through the processor's emulation port. The board's features include [2, p. 21]:

- ADSP-2181, 33 MIPS DSP.
- RS-232 interface.
- Socketed EPROM.
- Power supply regulation.
- Expansion connectors and user configurable jumpering.

After a great deal of development and tests was performed on the EZ-KIT Lite board, the final outcome seemed adequate, but upon implementation, the author experienced problems interfacing with the RS232 port of the RTU, and the interface component was damaged. Consequently the EZ-KIT Lite option was discarded.

This influenced the author to explore alternative options, resulting in the employment of another development board - a CME12-A4 by Axiom Manufacturing incorporating the Motorola MC68HC812A4 microcontroller.

4.6.2. The hardware assembly on the bread boarded single phase CME12-A4 commercial microcontroller development board

The Axiom CME12-A4 single board computer is a fully assembled, fully functional development system for the Motorola 68HC12A4 microcontroller. It is a commercial microcontroller development board, as shown in figure 4.5, and the main reason for using this controller was its speed and processing power. This

board contains a full 16 bit processor that can access 4MB program memory and 1MB of data memory.

The application as developed was efficiently written in Forth and required less than 32kB of program memory and very little RAM. The main components of the single phase prototype bread-boarded version, using the CME12-A4 measuring only one phase, consisted of:

- The input circuitry, as described in figure 4.4, incorporating the filter circuitry (LTC1063); the decoders (S44HC139N); the amplifier (LT1013); the oscillator and comparator (LM311N) circuitry.
- The Analogue to Digital part; the ADC (AD7864 by Analogue Devices).
- The initial single phase development board - a CME12-A4 by Axiom Manufacturing.

Figure 4.5 shows the layout on the single phase bread boarded version, using the CME12-A4 development board. Most of the development and laboratory tests were performed on this board.

A remarkable turn up found prior to using the Axiom CME12-A4 development board was that, while working on a 68HC12A4 microcontroller, initial tests with a low-cost microcontroller quickly revealed performance limitations in this application. However, those initial tests helped the author to determine the

processing power that was needed for the final product, and the Motorola MC68HC812A4 was selected for the final three phase board.

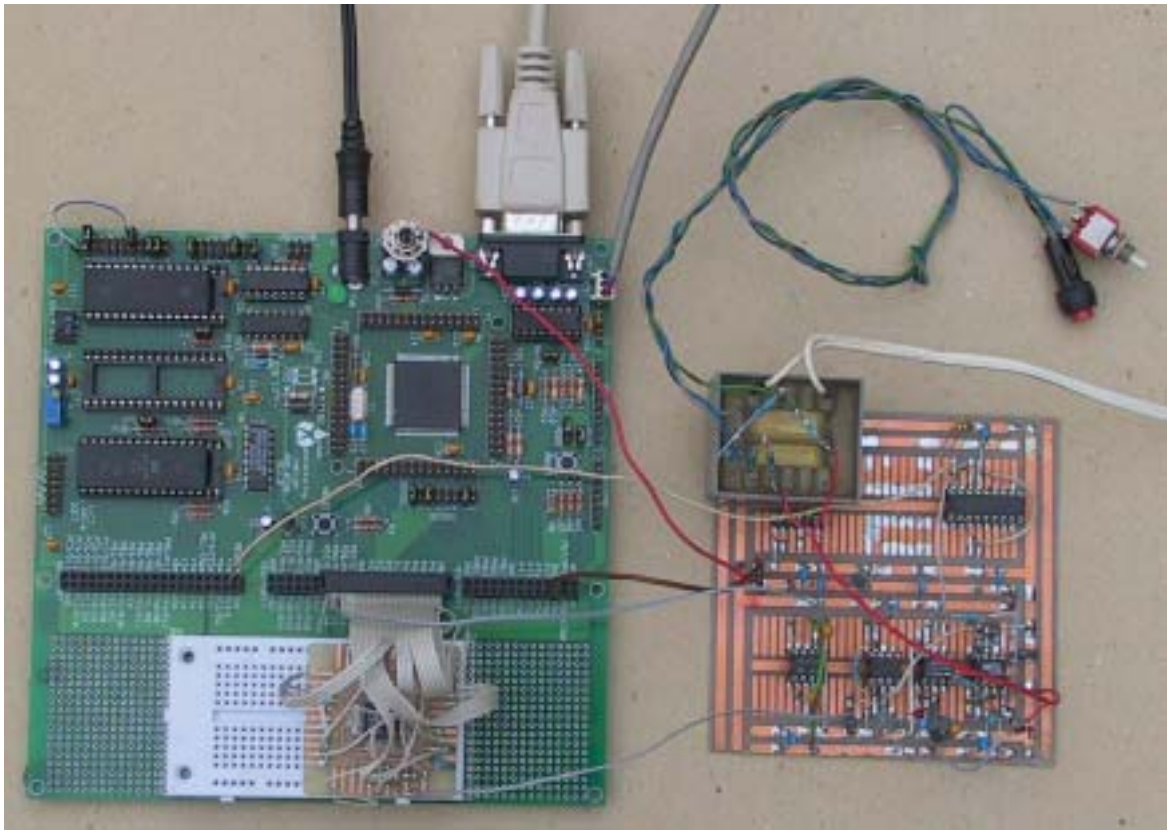


Fig. 4.5. The single phase bread boarded version and the CME12-A4 board.

The MC68HC812A4 is a 16-bit device with on-chip peripheral modules like.

- 16-bit CPU.
- Two asynchronous serial communications channels.
- Serial peripheral interface; timer and pulse accumulator.
- 8-bit ADC.
- 1 kbyte RAM; 4 kbyte EEPROM.
- Memory expansion logic.
- Key wakeup ports.

- ❖ Phase locked loop.

The external data bus can be 8 or 16 bits wide. On-chip memory mapping allows for more than 5 Mbytes address space. The maximum crystal frequency is 16MHz giving a basic instruction cycle time of 8MHz or 125ns.

Normal instructions take from 1 to 6 cycles, while multiply, divide, and multiply and accumulate take from 10 to 13 cycles.

4.6.3. The hardware assembly of the final three phase MC68HC812A4CPV8

The final layout and complete three phase circuit board was then manufactured according to the specifications and operation specified by the author, based solely on the single phase bread-boarded version discussed so far.

Figure 4.6 shows the final three phase board layout.

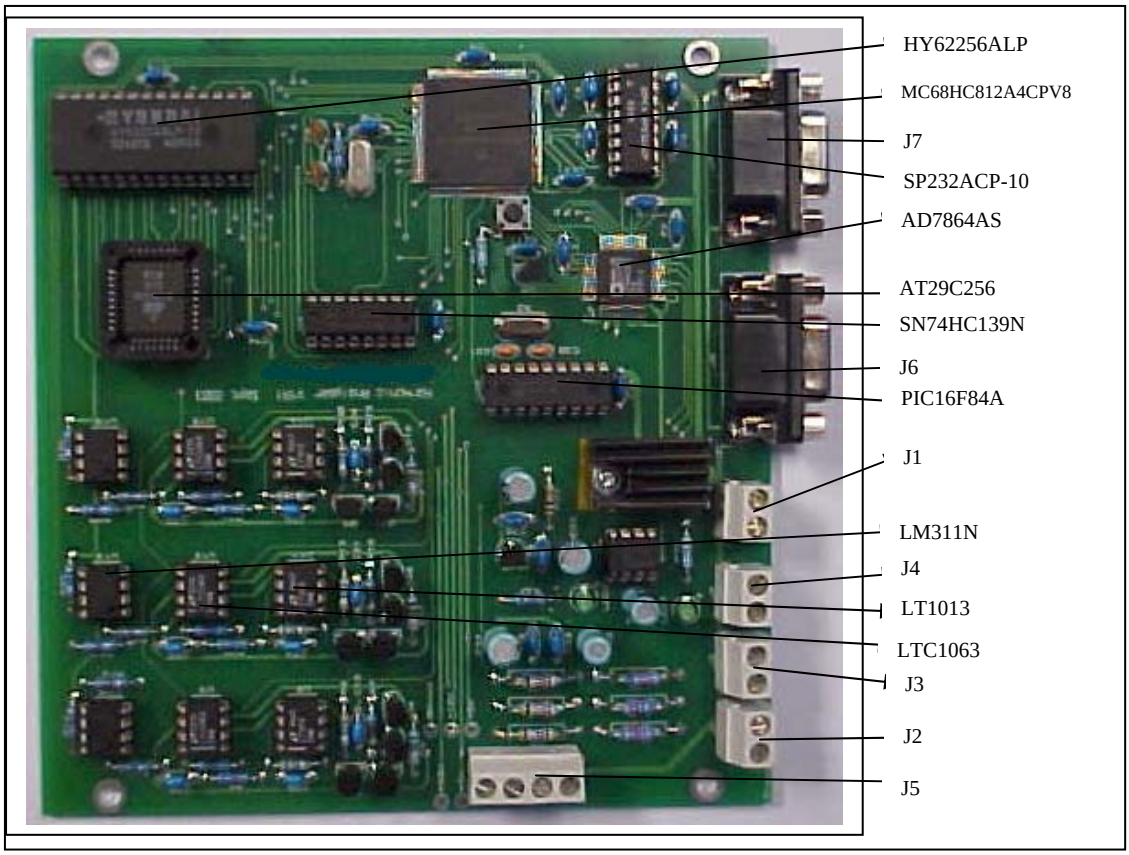


Fig. 4.6. The final IED quality of supply measuring device, using the MC68HC812A4CPV8 microcontroller.

- HY62256ALP –The HY62256A is the high-speed, low power and 32,786 x 8-bit CMOS Static Random Access Memory. It has a data retention mode that guarantees data to remain valid at the minimum power supply voltage of 2.0 volt.
- MC68HC812A4CPV8 – This is the microcontroller, replacing the 68HC12A4 Microcontroller described in the prototype.
- J7 – Terminal block J7 is the connection block to the RTU QUICC card for the MODBUS protocol.
- SP232ACP - The SP232ACP is a 2X2 driver / receiver device.

- AD7864AS - The AD7864 is the high speed, low power, 4-channel simultaneous sampling 12-bit A/D converter as described in section 4.3, and operates from a single +5VDC supply.
- AT29C256 - The AT29C256 is a five-volt-only in-system Flash programmable and erasable read only memory (PEROM). Its 256K of memory is organized as 32,768 words by 8 bits.
- SN74HC139N – The SN74HC139N comprises two individual 2 - 4 line decoders in a single package, designed for high-performance memory-decoding or data-routing applications requiring very short propagation delay times. This means that the effective system delay introduced by the decoders is negligible [7, p.1].
- J6 – This is the engineering port and may be connected to a computer to read harmonic and dip/surge information from the Microcontroller, and to reprogram the flash memory.
- PIC16F84A – This is an 18-pin Enhanced FLASH/EEPROM 8-bit Microcontroller which in this case was used as an oscillator, as the specific crystal that was needed at the time for the filter clock was not available and the PIC was used as a cost effective alternative to divide to the specific frequency.
- J1 – This is the 12VDC supply input terminal, and is not reverse polarity protected, and a 12VDC ±5% is acceptable.
- LM311N Voltage Comparator – This is a voltage comparator that has input currents nearly a thousand times lower than devices like the LM106 or LM710.
- J4 , J3, J2 are marked A, B and C. The software is triggered by the signal on A, so this must be connected first if there are less than 3 signals connected.
- LT1013 – This is a precision dual operational amplifier.

- LTC1063 – This is the DC accurate, clock-tuneable 5th order Butterworth Low pass filter.
- J5 – This terminal block is for signal input testing.
- LED1 – Indicates 12VDC power supplies are working.
- LED2- Indicates -5VDC power supplies are working.

After the development and manufacturing of the final IED circuit board, the final configuration and performance of the IED card, concurrently with the RTU QUICC card was evaluated and tested. This incorporated the UNICON configuration, as well as the communications part. The final configuration setup is explained by figures 4.17 and 4.20.

4.7. The IED positioning on the RTU

If one looks at the final three phase IED development in figure 4.6, it will be found that the author did the design in such a manner that the DB9 female connectors J6, (the engineering port connected to a computer to read harmonic and dip/surge information from the microcontroller) and J7 (the connection block to the RTU QUICC Card for the MODBUS protocol) have identical PCB connections, as shown in table 4.1. The software is described in section 4.11. The IED was mounted in a horizontal position on the RTU sub rack, beneath the vertical QUICC and MTIO card. The dimension of the IED was specifically designed so that the card may fit here to avoid the installation of an additional panel to house the IED device [Fig. 4.7] to the RTU.

Table 4.1. Pin connections of IED J6 and J7.

Pin no.	Function
2	Transmit from J
3	Receive to J
5	Ground
1; 4; 6	Connected together
7; 8	Connected together

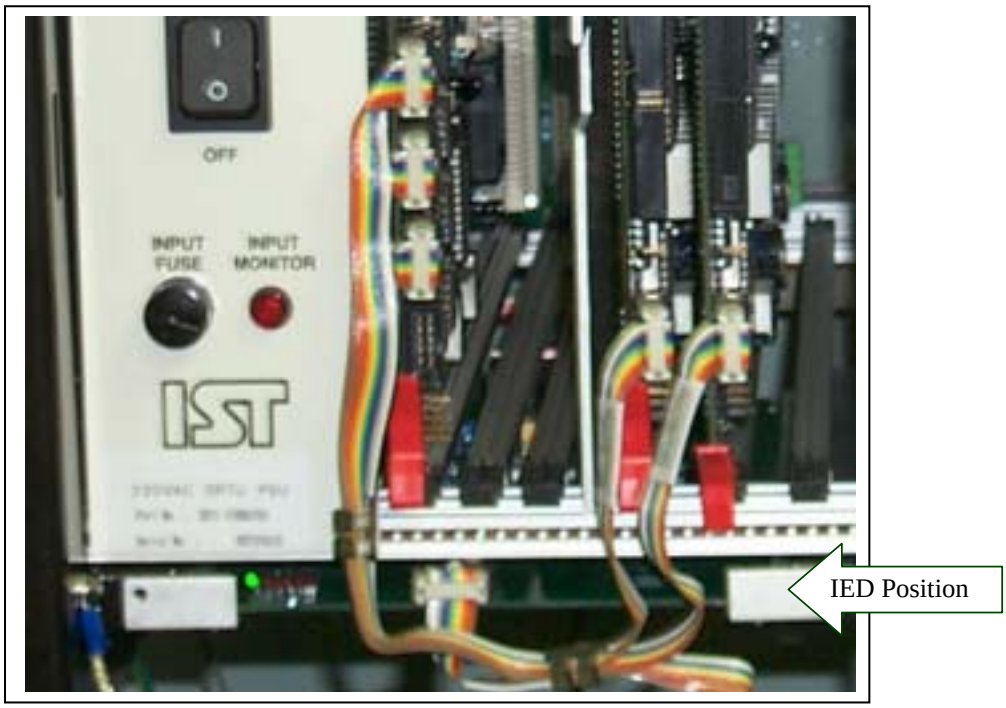


Fig. 4.7. Position of the IED measuring device on the RTU.

4.8. IED Software development

This section describes the software configurations as explained in figure 4.1 that was followed during the prototype development phase. A good number of software developments were done during the ADSP-2181 EZ-KIT Lite stage, and later implemented to the CME12-A4 development breadboard for testing purposes. To be able to prove accuracy, a simulation test suite was then developed under LINUX.

4.8.1. Mathematical wave generation using LINUX

On a Pentium PC, the C compiler of LINUX has a library function to generate sine values for simulation purposes. With that in mind, a program called *wave.cpp* was written to simulate the mains condition that the author was trying to monitor. *Wave.cpp* is a "collection of harmonics" which are added together and output as 16-bit samples, each sample representing a period of time. The amplitude of each harmonic can be specified. The following code fragment from the main function illustrates setting harmonics:

4.8.1.1. Harmonic simulations

```
main()
{
  /* set the amplitude of the harmonics wanted */
  /* set one frequency to 100 (fundamental, usually) and
  then the others to a percentage of that.  A scaling
  factor to avoid overflow will be calculated later */

  harmonic_amplitude[0]=100;
  harmonic_amplitude[4]=8;
  harmonic_amplitude[8]=22;
  harmonic_amplitude[12]=4;
```

In this case, the amplitude of the fundamental is set to 100. 100 is a convenient number for the fundamental, as all numeric amplitudes of subsequent harmonics become a percentage of this value. In this case, the 5^{th} harmonic is set to 8% of the fundamental, the 9^{th} to 22% and the 13^{th} to 4%. The code is zero-based [24, p. 75], so the fifth harmonic occurs at array index 4.

The *wave.cpp* program allows up to 16 harmonics (defined as NUM_HARMONICS) and this is doubled to give the sampling rate. These values are an exact power of 2, as per text-book recommendations. On the *wave.cpp* program, a 50Hz sine wave sampled 32 times per cycle translating to a sampling period of 625 microseconds. This program outputs data samples in 16-bit hex format. Each sample value is the value that would occur 625 microseconds after the last.

4.8.1.2. Dip simulations

This part introduces the voltage dips to our standard 50 Hz wave.

A voltage dip is a modulation of the amplitude envelope of the mains waveform [Ch.2, pp. 17, 18], and to simulate this, a program was written in LINUX – called "*filter.c*" [App. 1, p. 253]. The sole purpose of this program was to modify the amplitude of the samples that pass through it. When placed after the *wave.cpp* program, it can be used to reshape the output of *wave.cpp* with the required dip information. Dips could therefore be easily inserted or removed from the simulation chain. *Filter.c* reads dip information from a file specified as a command-line argument.

The file contains the filtering profile information in a simple language that is defined as follows:

- Lines starting with # are ignored – they are used for comments.
- Lines starting with a non-digit are assumed to be garbage and are also ignored.
- 75 27 means adjust amplitude to 75% on the next 27 samples.
- 55 means adjust amplitude to 55% on the next sample.

The filter can adjust the first 10 000 samples, after which there is no more filtering as shown in figure 6.4, page 184. This shows how this file has modified a set of samples.

4.8.1.3. Coefficient determination

The software was required to detect harmonic components on a 50Hz sinusoidal input wave up to the 13^{th} harmonic. The frequencies that were needed to be analysed for this purpose are therefore 50; 100; 150; 200; 250; 300; 350; 400; 450; 500; 550; 600; and 650Hz.

4.8.1.4. Selecting a sample frequency

The principle behind DSP is to take snapshot samples at various points along a wave, and to do calculations on those samples. The skill is to make sure that your samples are a realistic representation of the wave you are trying to analyse. Obviously, the more samples you take per cycle, the better your representation of the wave, but now you need more processing power to work through the extra data. During the research it was attempted to get a balance.

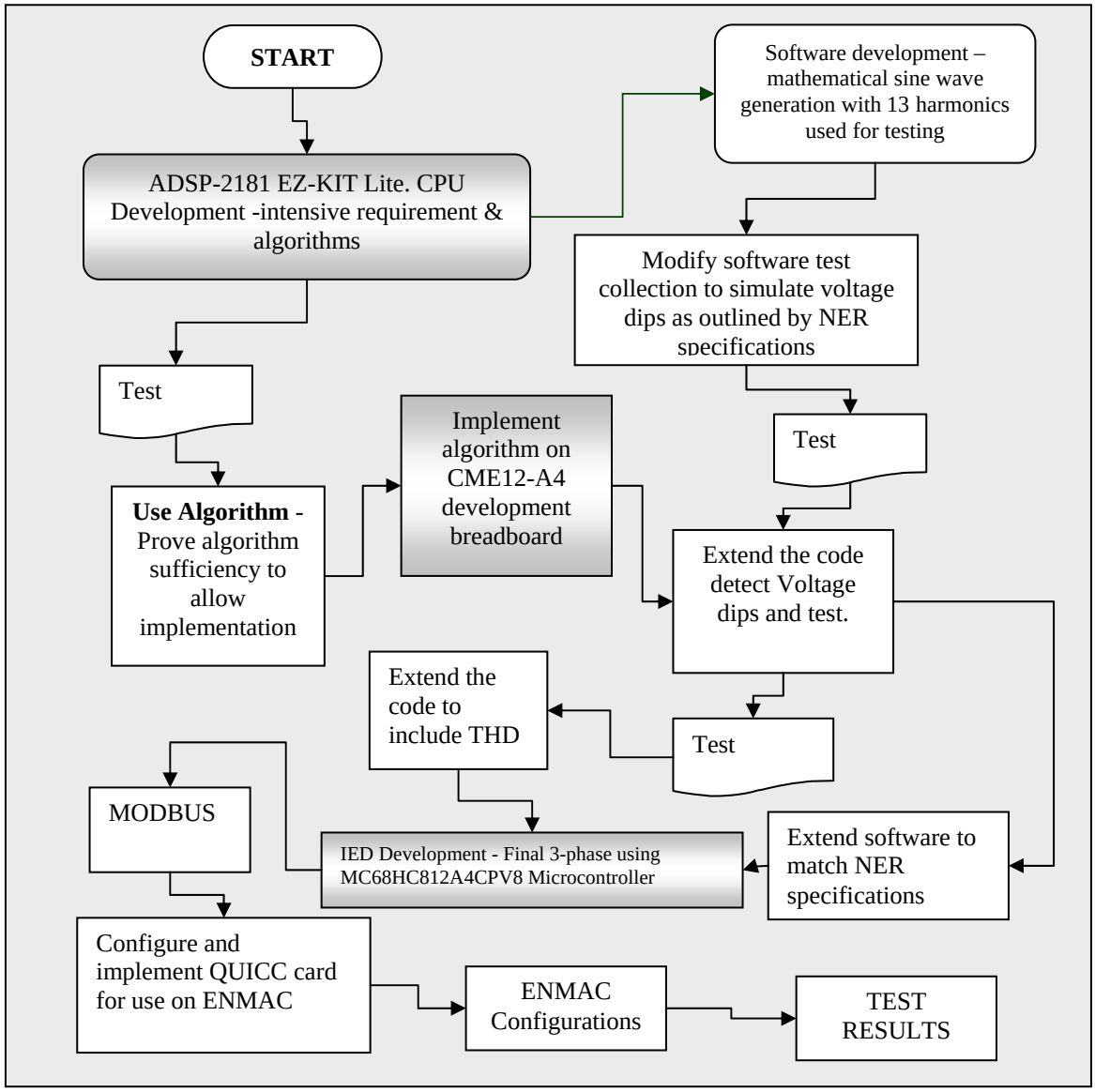


Fig. 4.8. Flow diagram of the chronological phases followed during the software development.

The Nyquist theory tells us that you need at least two points per cycle on a sine wave in order to be able to detect it [6, p. 35]. The sampling frequency has to be greater than twice the Nyquist frequency or, equivalently, twice the bandwidth of the signal being sampled. This means that a higher sampling rate allows you to detect higher frequency harmonics. Furthermore, the exact choice of sampling rate favours certain frequencies. For instance, a sampling rate of 400 Hz is high enough to detect waves of both 50 Hz and 49 Hz, as the Nyquist criteria were

met in both cases. But, because the sampling frequency is an exact multiple of 50Hz, "more" information can be derived from a 50Hz sample than from one at 49Hz. In many cases (like digital audio), you need to cater for a wide variety of different frequencies, so you cannot favour one in particular. But in this special case of harmonic detection, one can.

The author knew in advance the exact frequencies that we were concerned with, so our sampling rate could be optimised to the frequencies we were interested in. Since the highest harmonic that the author was interested in - the 13^{th} harmonic - has a frequency of 650Hz, we knew that according to the Nyquist theory you required a sampling frequency of at least twice this - i.e. 1300Hz.

To optimally select the best sampling frequency for the harmonics we were interested in, the author started by splitting each harmonic into its prime factors.

Table 4.2. Table showing the prime factors of the sample frequencies.

Frequency	Factors	Factors	Factors	Factors	Factors
50 factored out	2		5 5		
100 factored out	2 2		5 5		
150 factored out	2	3	5 5		
200 factored out	2 2 2		5 5		
250 factored out	2		5 5 5		
300 factored out	2 2	3	5 5		
350 factored out	2		5 5	7	
400 factored out	2 2 2 2		5 5		
450 factored out	2	3 3	5 5		
500 factored out	2 2		5 5 5		
550 factored out	2		5 5		11
600 factored out	2 2 2	3	5 5		
650 factored out	2		5 5		13

For the purpose of this development, the author fortunately has a special case; all harmonics were exact multiples of the fundamental. So, once the author found the best frequency for the 13^{th} harmonic, it was bound to be good for all other harmonics. Once the author had a list of frequencies to help selecting the best sampling frequency, two C programs were used (*factor.c* and *primes.c*) to factor each frequency into its prime factors. Once each frequency was broken down into its constituent prime factors, the prime factors that were most common to the greatest number of frequencies were selected, and then multiplied together. Call the resulting product "A", we knew now that the best sampling frequency is an integer multiple of "A" that is greater than or equal to twice the highest input frequency of interest.

DSP theory tells us that the best sampling frequency is an integer multiple of the product of the most common prime factors in the input sample [2, p. 481]. Using this method, some tested sample frequencies were:

- 2 x 650 = **1300**
- 2x2x5, scaled up to (1300) = **1320**
- 2x3x5x5, scaled up to (1320) = **1470**
- 2x2x2x3x5x7, scaled up to (1470) = **2310**
- 2x2x2x3x3x5x7x11, scaled up to (2310) = **30030**

The *bestfs.c* program is a C program that verifies if the chosen sampling frequency is the best fit. The author got the program from the DSP suppliers themselves, but made small modifications to suit our needs. The program was run on each individual tone in the tone set - in our case, on each harmonic, and

generated a maximum error-squared value to give an indication as to how well the sampling frequency matches the frequency under test.

The *bestfs.c* program sweeps through the specified range of sampling frequencies and calculates the maximum squared mismatch error for the tone set. The mismatch error is an indication of how closely the tone of interest matches the integral subdivisions of the sampling frequency. Frequencies with many common prime factors will match more precisely.

For a given tone set, the mismatch error-squared is calculated for each individual tone. The largest mismatch in the tone set is chosen as the maximum error-squared value for the tone set at that sampling frequency. This is the term that should be minimized.

All of these sampling frequencies are good candidates for meeting the criteria laid down, but DSP theory also indicates that the best frequency is the one with the lowest mismatch error-squared [2, p. 482]. Tables 4.3 - 4.7 show the outputs of the *bestfs.c* program that calculates the error-squared for each of the suggested frequencies above, plus a window of 50Hz on either side, stepping in increments of 10Hz. The window is intended to give a representation of what would happen if the mains frequency drifts.

Tables 4.3 to 4.7 show the output of the program:

Table 4.3. Frequency 1250.0 Hz to 1350.0 Hz: stepping 10.0 Hz.

Nyquist violation: f_freq=650.0	at f_sampling=1250.0 (10Hz step)
0.250000 = maxerrsqr	(at f_sample = 1250.0)
0.230400 = maxerrsqr	(at f_sample = 1260.0)
0.217777 = maxerrsqr	(at f_sample = 1270.0)
0.217777 = maxerrsqr	(at f_sample = 1280.0)
0.202500 = maxerrsqr	(at f_sample = 1290.0)
0.250000 = maxerrsqr	(at f_sample = 1300.0)
0.202500 = maxerrsqr	(at f_sample = 1310.0)
0.160000 = maxerrsqr	(at f_sample = 1320.0)
0.187778 = maxerrsqr	(at f_sample = 1330.0)
0.217778 = maxerrsqr	(at f_sample = 1340.0)
0.250000 = maxerrsqr	(at f_sample = 1350.0)

Table 4.4. Frequency 1270.0 Hz to 1370.0 Hz: stepping 10.0 Hz.

Nyquist violation: f_freq=650.0	at f_sampling=1270.0 (10Hz step)
0.217777 = maxerrsqr	(at f_sample = 1270.0)
0.217777 = maxerrsqr	(at f_sample = 1280.0)
0.202500 = maxerrsqr	(at f_sample = 1290.0)
0.250000 = maxerrsqr	(at f_sample = 1300.0)
0.202500 = maxerrsqr	(at f_sample = 1310.0)
0.160000 = maxerrsqr	(at f_sample = 1320.0)
0.187778 = maxerrsqr	(at f_sample = 1330.0)
0.217778 = maxerrsqr	(at f_sample = 1340.0)
0.250000 = maxerrsqr	(at f_sample = 1350.0)
0.223471 = maxerrsqr	(at f_sample = 1360.0)
0.240992 = maxerrsqr	(at f_sample = 1370.0)

Table 4.5. Frequency 1420.0 Hz to 1520.0 Hz: stepping 10.0 Hz.

Nyquist violation: f_freq=650.0	at f_sampling=1270.0 (10Hz step)
0.217777 = maxerrsqr	(at f_sample = 1420.0)
0.217777 = maxerrsqr	(at f_sample = 1430.0)
0.160000 = maxerrsqr	(at f_sample = 1440.0)
0.250000 = maxerrsqr	(at f_sample = 1450.0)
0.187778 = maxerrsqr	(at f_sample = 1460.0)
0.202500 = maxerrsqr	(at f_sample = 1470.0)
0.217778 = maxerrsqr	(at f_sample = 1480.0)
0.233611 = maxerrsqr	(at f_sample = 1490.0)
0.250000 = maxerrsqr	(at f_sample = 1500.0)
0.233611 = maxerrsqr	(at f_sample = 1510.0)
0.217778 = maxerrsqr	(at f_sample = 1520.0)

Table 4.6. Frequency 2260.0 Hz to 2360.0 Hz: stepping 10.0 Hz.

Nyquist violation: f_freq=650.0	at f_sampling=2260.0 (10Hz step)
0.230400 = maxerrsqr	(at f_sample = 2260.0)
0.242367 = maxerrsqr	(at f_sample = 2270.0)
0.242367 = maxerrsqr	(at f_sample = 2280.0)
0.227456 = maxerrsqr	(at f_sample = 2290.0)
0.250000 = maxerrsqr	(at f_sample = 2300.0)
0.202500 = maxerrsqr	(at f_sample = 2310.0)
0.217777 = maxerrsqr	(at f_sample = 2320.0)
0.217777 = maxerrsqr	(at f_sample = 2330.0)
0.160000 = maxerrsqr	(at f_sample = 2340.0)
0.250000 = maxerrsqr	(at f_sample = 2350.0)
0.193600 = maxerrsqr	(at f_sample = 2360.0)

Table 4.7. Frequency 29980.0 Hz to 30080.0 Hz: stepping 10.0 Hz.

Nyquist violation: f_freq=650.0	at f_sampling=29980.0 (10Hz step)
0.240992 = maxerrsqr	(at f_sample = 29980.0)
0.223472 = maxerrsqr	(at f_sample = 29990.0)
0.206610 = maxerrsqr	(at f_sample = 30000.0)
0.190413 = maxerrsqr	(at f_sample = 30010.0)
0.174876 = maxerrsqr	(at f_sample = 30020.0)
0.160020 = maxerrsqr	(at f_sample = 30030.0)
0.159995 = maxerrsqr	(at f_sample = 30040.0)
0.250000 = maxerrsqr	(at f_sample = 30050.0)
0.159995 = maxerrsqr	(at f_sample = 30060.0)
0.217771 = maxerrsqr	(at f_sample = 30070.0)
0.217771 = maxerrsqr	(at f_sample = 30080.0)

The tables verify that with a sampling frequency suitable for the 13^{th} harmonic, the error was small for all harmonics. Since the lowest values of error-squared give the best performance, the following possibilities were extracted:

1) 0.159995 = maxerrsqr (at f_sample = 30040.0)

2) 0.159995 = maxerrsqr (at f_sample = 30060.0)

3) 0.160000 = maxerrsqr (at f_sample = 1320.0)

4) 0.160000 = maxerrsqr (at f_sample = 1440.0)

5) 0.160000 = maxerrsqr (at f_sample = 2340.0)

6) 0.160020 = maxerrsqr (at f_sample = 30030.0)

7) 0.174876 = maxerrsqr (at f_sample = 30020.0)

These seven frequencies will hereafter be referred to as the frequencies under consideration. This error-squared output indicates that a sampling frequency of 30040Hz best suit the frequencies given, but a much lower sampling frequency of 1320Hz is nearly as good. But before deciding on a sampling frequency, three other factors also had to be considered:

- **Leakage loss**: To minimise leakage loss, the sampling frequency had to be an exact multiple (or close to it) of the frequency under consideration. The frequency of 30030.0Hz above had best meet the requirement, as it was close to an exact multiple of all of the frequencies we were interested in; 50, 100, and 150 up to 650Hz.
- **Frequency resolution**: The higher the freq. rate, the better the frequency resolution and therefore the better the noise rejection.
- **Detection time**: The number of samples per cycle should not exceed the processing power of the DSP in order to maintain real-time performance.

Taking all of these factors into consideration, the author selected the highest frequency from the list under consideration that can be handled by the processing power of the DSP. Since the Goertzel algorithm used requires about 120 cycles, at the processor speed of 33MHz, this equates to a loop time of just less than 4 µSec, as explained below:

At 33MHz - 1 cycle takes (1/33000000) sec.

Therefore for 120 cycle's = (120/33000000) sec.

=3.6 µSec.

At a sampling frequency of 30030Hz, the period is about 33 µSec, therefore at 30030Hz; one had to sample every 33 µSec - i.e. 1/30030. Thus the sampling rate at 30030Hz was not too high for the processor.

4.8.1.5. Choosing the number of samples, *N*, per loop

As we know by now, the Goertzel algorithm works on a sample-by-sample basis. After N iterations or N samples have been received, a value is output. From the calculations performed above, if N is 1, the DSP could generate an output approximately every 4 microseconds. This would be pointless - firstly, too much data would be generated at the output and secondly, there would be about 8 outputs per sample. That is, 8 outputs of the same value.

As *N* is increased, the resolution of data improves, but the longer it takes before the information becomes available. The value of *N* has nothing to do with real-time processing, and it is quite acceptable for the loop to take longer than the sampling frequency period, as it does not affect real-time capability.

In our application, producing an output every second is fast enough. Since a cycle takes 33 microseconds, this means that N should be set to about 30000. But just as the sampling frequency can be adjusted to suit particular frequencies,

so can N also be adjusted. Setting N to 30030 instead of 30000 gives a lower error-squared value, because it is an exact multiple of the frequencies to be detected. This optimisation can be done by performing calculations over a range of frequencies and choosing the lowest error-squared result that is calculated. We did this using the *bestn.c* program shown in appendix 1.10 [App.1, pp. 263], with the following results:

0.249481 =maxerrsqr (N= 30000) detect= 999.001 ms resolu = 1.001 Hz

Setting N to 30030, giving exactly 1sec. of resolution yields the following:

0.000000 =maxerrsqr (N= 30030) detect = 1000.000 ms resolu = 1.000 Hz

Having chosen a value for the sampling frequency and for N, the *Coefgen.c* program was then used to calculate the DSP coefficients. [App. 1, p. 265]. The algorithm, as it stands, has no provision to output the results, and had to be modified. As the tones to be detected were all multiples of one frequency, it was expected that the DSP at that stage had more than enough processing power to cope. It was estimated that less than 10% of the CPU was actually needed. A PC was then used to mathematically generate a set of samples, thereby eliminating the need for the SABS test services.

Upon interfacing the ADSP-2181 EZ-KIT Lite to the RTU, the port got damaged. On arrival of the new CME12-A4 development board, these generated samples were then injected into the program to test proper operation on the prototype bread boarded CME12-A4.

4.8.1.6. Dip detection routine

The routine developed for harmonic detection looks at a frame of samples (representing one period) and calculates the amplitude content, which it then reports. To cut processing time to a minimum, the dip-detection routine uses the value that is already calculated for the fundamental frequency, and works with that. A "sliding scale" window keeps track of the maximum value (continually adjusted over time) calculated for the fundamental**,** as this represents the wave voltage when there are no dips.

As soon as a harmonic algorithm reports a value that is <90% of this value, the dip detection routine suspects the start of a dip and records the incoming samples for comparison. Up to 150 samples are recorded and as soon as the dip finishes (the amplitude jumps up to >90% of the peak), the sample train is analysed to determine the type of dip. According to the NER specifications of "dip types", small dips <10% are ignored [Ch.3, p. 38].

A number of tests were devised and fed into the *filter.c* program, which was found to perform satisfactorily. There are, however, some limitations worth noting:

- Since this algorithm uses components from the harmonic detection routine, it also requires that the sampling rate be an exact fraction of the period.
- This algorithm uses only the fundamental wave in its calculation, so is largely immune to harmonic distortions – which is good!

- The algorithm operates on one frame at a time, which can result in some "blurring" of the edges of the dip detection. A frame can be defined as a number of 20ms cycles. If a group of 150 cycles is continuously monitored in a sliding-window (circular buffer) fashion, the frame is big enough to detect all the dip types that we are interested in.

4.9. IED software configuration

The DSP Software design for the Goertzel algorithm was required to detect harmonic components on a 50Hz sinusoidal input wave, and it was decided that for purpose of implementation only up to the 13th harmonic would be tested. The information supplied by Babst [2, pp. 481-502] for implementing the Goertzel Algorithm [Ch.2.10, p. 32] in an application like that needed for this research was used, as the algorithm uses fast polynomial expansions to give both accuracy and flexibility with minimal processing requirements. The coefficients of the algorithm needed to be calculated for the frequencies of interest to us, and then tested for suitability. The program is presented in appendix 1. (App.1, p. 240]. This program was used to generate the tests, *bestfs.c*, and gave a perfect fit to the Goertzel algorithm. The algorithm, as it stands, has no provision to output the results, and the program needed to be adapted so that it could run on a suitable DSP board.

As mentioned in the hardware section 4.6.2, the processor software was written in Forth, using a software development package from Forth Inc., USA. It allows efficient multi-tasking and full control over the processor. There are essentially four sections to the software program:

- Data gathering.
- Harmonic calculation.
- Dip/surge analysis.
- Data output.

Data gathering is handled entirely using hardware interrupts, so it takes priority over any higher level programming. This is a robust configuration that ensures that processing takes place on reliably-sampled data. The zero-crossing signal starts the measurement of 64 samples, taken symmetrically over one cycle. The data is moved to buffers at the end of each cycle for use by the harmonic calculation task. While taking the samples the sum of the squares is calculated to give an RMS value per cycle for use by the dip/surge analysis task. Please note that the inclusion of zero-crossing hardware guarantees proper synchronisation at all times - the "drift" problem noted in the software simulation results on table 6.6, page 181 has thus been eliminated completely.

The **harmonic calculations** are based on the algorithms as discussed so far. The algorithms used cross correlation, to minimise processing power at the expense of slightly more memory (of which we have a sufficient amount). This technique compares the signal samples with stored lookup tables, rather than calculating the values each time. The magnitude of the first 13 harmonics is calculated, as well as the total harmonic distortion, RMS, and peak values. This is a performance critical section of the software, and has been suitably optimised - this cycle of calculations

takes nearly one second as a single task and up to two seconds if the other higher priority tasks are running as well.

The **dip/surge** calculation takes the per-cycle RMS value at the end of each cycle and compares it with previous cycles. Dips and surges are classified according to the Voltage dip Window of the NRS 048-2:1998 [Ch.3, p. 38]. Out of limit data can be outputted in real time if necessary, but requires a fast serial connection and equipment that can store the bursts of data.

The **data output** interface has been decided on as a serial link using MODBUS format; port 4. The software at this stage has been written on the RTU side, but, not having communicated formally with commissioned field equipment; it was only tested on the FieldComm Test Set, as described in chapter 6.6.1 [Ch.6, p. 187].

4.10. Harmonic distortion calculation

The conventional approach to finding the frequency content of an unknown signal would be to use the Fourier transform [Ch.2, p. 31]. However, in our particular situation the problem is much simplified in that known frequencies are being measured. The fundamental signal is 50Hz and the harmonics to be measured are multiples of 50Hz. This special case was done best by using cross correlation [38, p. 157].

Therefore, each sample in the frequency domain was found by multiplying the time domain signal by the sine or cosine wave being looked for, and adding the resulting

points [38, p.158]. The single number that resulted from this procedure was a measure of how similar the two signals are.

The author was looking for 13 fixed pure sine frequencies in the mains signal with harmonics from 50 to 650 Hz, where the harmonics are not necessarily exactly in phase with the fundamental. Therefore, the author looked for both sine and cosine frequencies, and then got their combined magnitude, according to the formulae [38, p. 158]:

$$\mathrm{Re}\,X[k] = \sum_{I=0}^{N-1} x[i]\cos(2\pi ki / N) \quad \text{----------(4.1)}$$

$$\mathrm{Im}\,X[k] = \sum_{I=0}^{N-1} x[i]\sin(2\pi ki / N) \quad \text{----------(4.2)}$$

$$MagX[k] = \sqrt{(\mathrm{Re}\,X[k]^2 + \mathrm{Im}\,X[k]^2)} \quad \text{----------(4.3)}$$

Where:
Re stands for REAL, *Im* stands for imaginary and *Mag* stands for magnitude.
$x[i]$ is the time domain signal being analised.
Re$X[k]$ & Im$X[k]$ are the frequency domain signals being calculated. The index I runs from 0 to N-1, while the index k runs from 0 to N/2
k is the harmonic and N is the number of samples.

To do the calculations as fast as possible, integer arithmetic was used and the COS and SIN functions were compiled into lookup tables. As the phase of the fundamental was unimportant, the same tables could be used for all the mains phases. The resulting magnitudes were normalised so that the fundamental was 1000, i.e. 100.0%.

4.11. Software description for harmonics and voltage dip/surge calculation on the CME12-A4 for measuring harmonics and voltage dips

During this phase, the ADSP-2181 EZ-KIT Lite was replaced by the CME12-A4, and the LINUX programs developed for harmonics and voltage dip/surge measurements incorporated to the new CME12-A4 prototype were based on the algorithms discussed so far. It is executed by a command "GO", then read by the RTU via MODBUS to the communication port of the CME12-A4, and can be summarised as follows:

- **The power-up task** - At power-up, the routine {GO} is executed, which initialises parameters, creates tasks, starts the interrupt system and then runs the tasks.
- **The communications task** – This task looks for any MODBUS slave address and receive the rest of the messages (do it).
- **The dips and surges calculation** - During the cycle the 64 samples are squared and summed and put into a buffer. Then the dip program runs, which passes the R.M.S. value to a main calculation dip file. The results are moved to a buffer for the communications task to have access to. Then it checks the consecutive R.M.S. values. An event is created when the R.M.S. value compares with the specified error box. A flag is set when the event returns to normal; the minimum/maximum R.M.S. value and the duration of the dip/surge are recorded.

- **Harmonics calculation** – This file takes 1 cycle's worth of samples and puts it into an array for use by the harmonic calculations. It calculates the sine and cosine values of the harmonics & converts them into polar co-ordinates where only the magnitude is used. The harmonics are then normalised to a percentage.
- **THD calculation** – This program determines the square root of the sum of the square harmonics / sum of the square of fundamentals + harmonics.
- **RMS calculation** - The R.M.S. value is calculated by: square root ((sum of 64 values squared) / 64) and normalised to a peak value. The peak is the biggest amplitude either positive or negative of the 64 samples, and normalised so that a signal with a peak value of half the ADC range is represented by 100.0%. The results are moved to a buffer for the communications task to have access to.
- **The peak** is the biggest amplitude either positive or negative of the 64 samples, and normalised similar to the RMS calculation.

Figures 4.9 – 4.14 explain these programs, where any word in { } refers to the name of a routine in the source code.

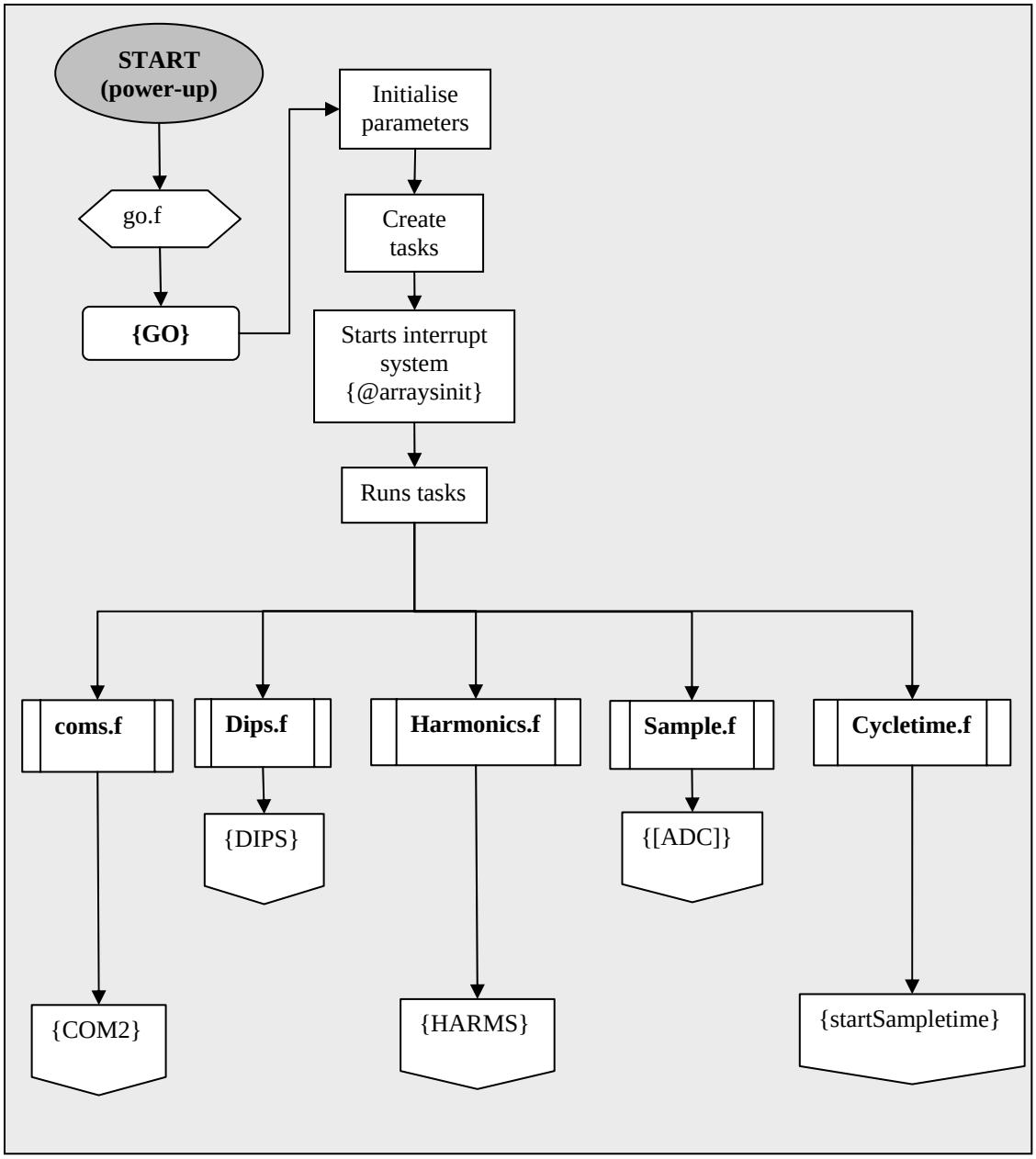


Fig. 4.9. Flow diagram of the program that is used on the CME12-A4 prototype, showing the process flow when the {GO} routine is executed.

On the next diagram, figure 4.10, the routine {COM2} runs in a loop, and is normally stopped waiting for an input through the serial port from the MODBUS network. Any received byte is checked to see if it is our MODBUS slave address.

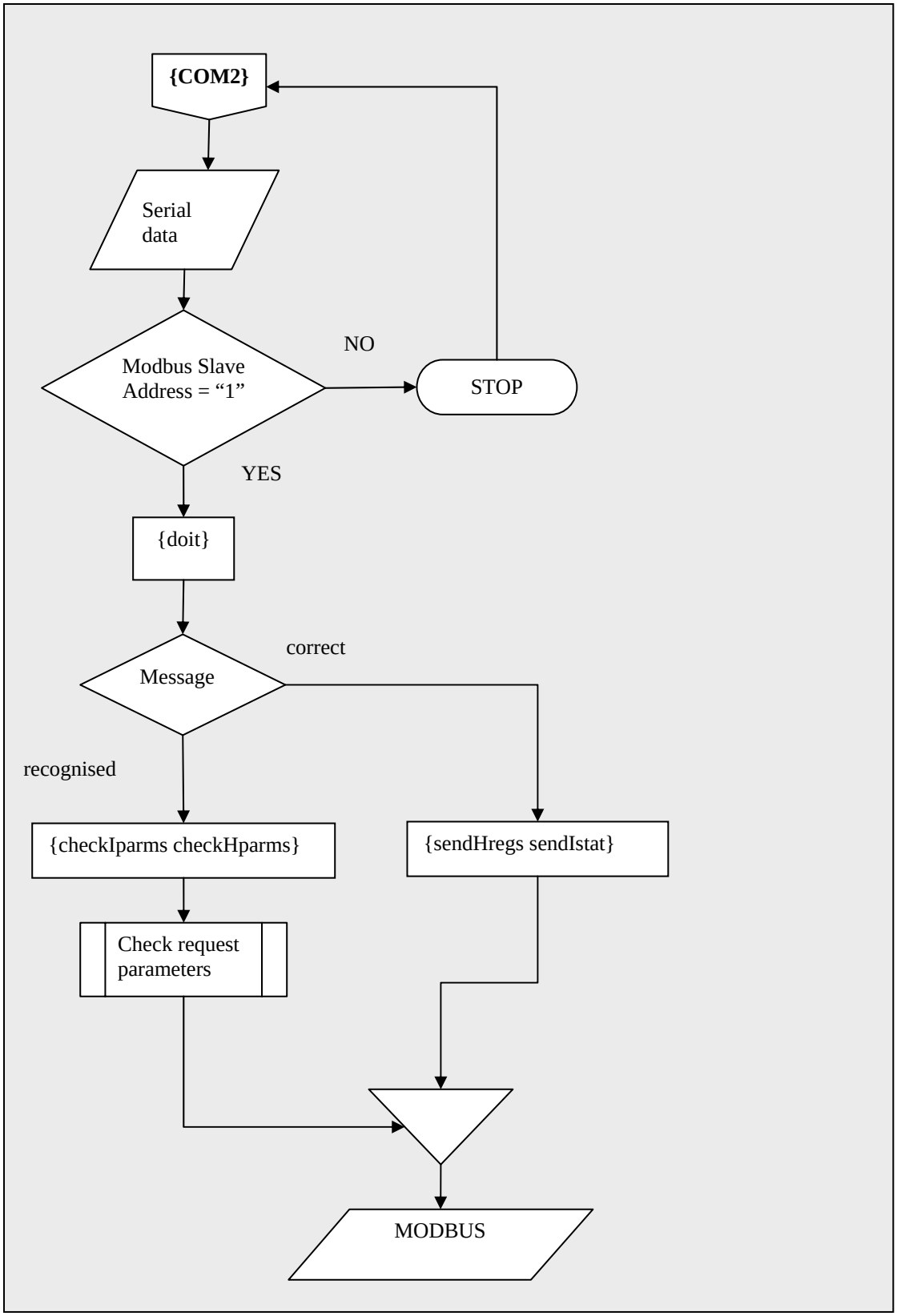


Fig. 4.10. Flow diagram of the communications task {COM2} on the CME12-A4 single phase prototype to enable MODBUS protocol message exchange between the IED and the RTU.

The MODBUS message with appropriate header, data, and CRC is described in chapter 3.7.5 [Ch. 3, pp. 72, 75]. The next flowchart (Fig. 4.11) describes the dip and surge calculation program.

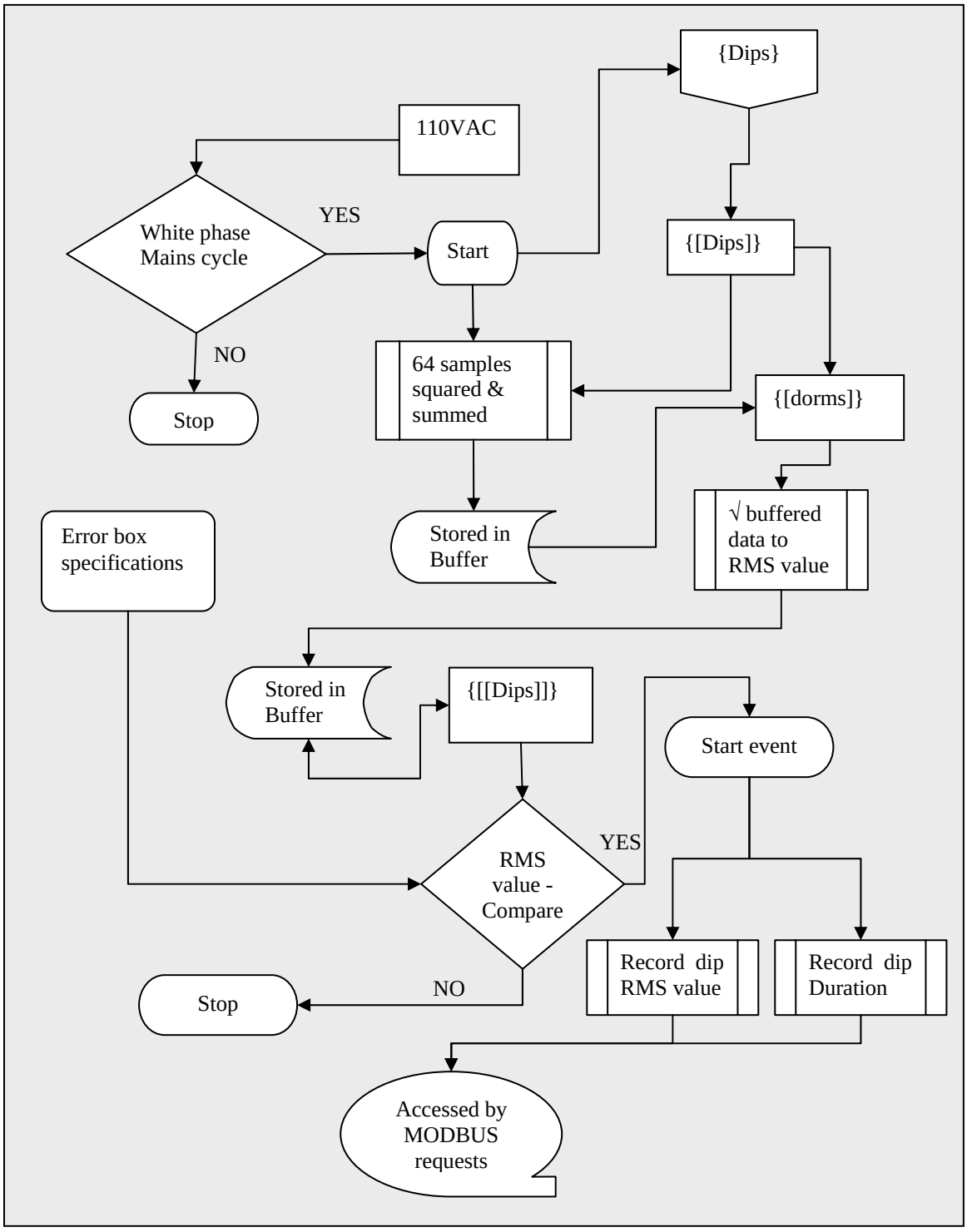


Fig. 4.11. Flow diagram of the dips and surge calculation task on the CME12-A4 single phase prototype.

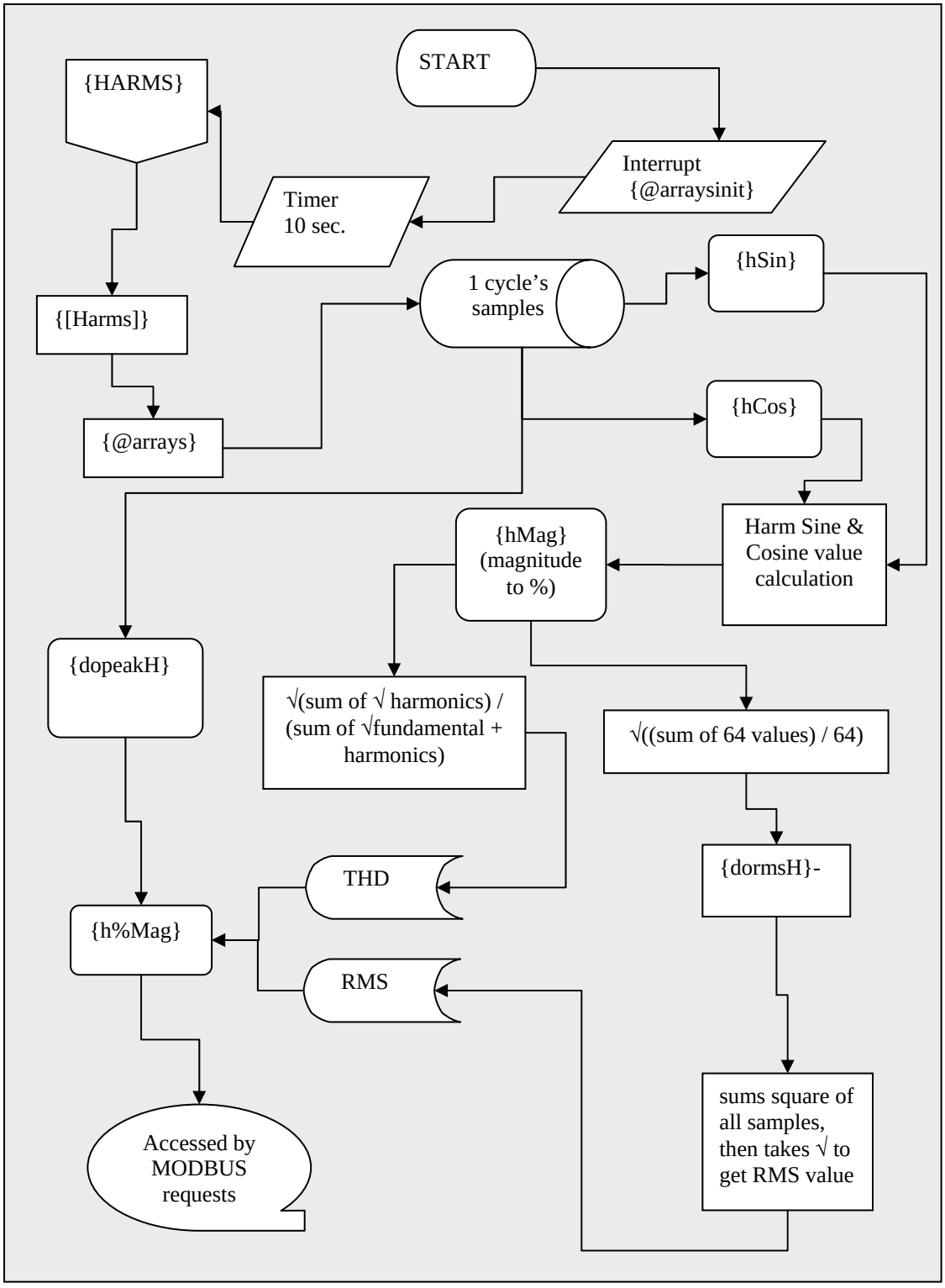


Fig. 4.12. Flow diagram of the harmonics calculation task on the CME12-A4 single phase prototype.

In figure 4.12 the harmonics calculation task {HARMS} does the calculation for only one of the three phases. The calculations for the other phases are identical. {hSin} and {hCos} calculate the sine and cosine values of the harmonics, where {dormsH} sums the square of all the samples, then takes the square root to get the R.M.S. value. The file {dopeakH} finds the highest absolute value of all the samples. {hMag} takes the square root of the sum of the squares of the sine and cosine results to get the magnitude. {h%Mag } normalises the fundamental harmonic to 1000, i.e. 100.0%.

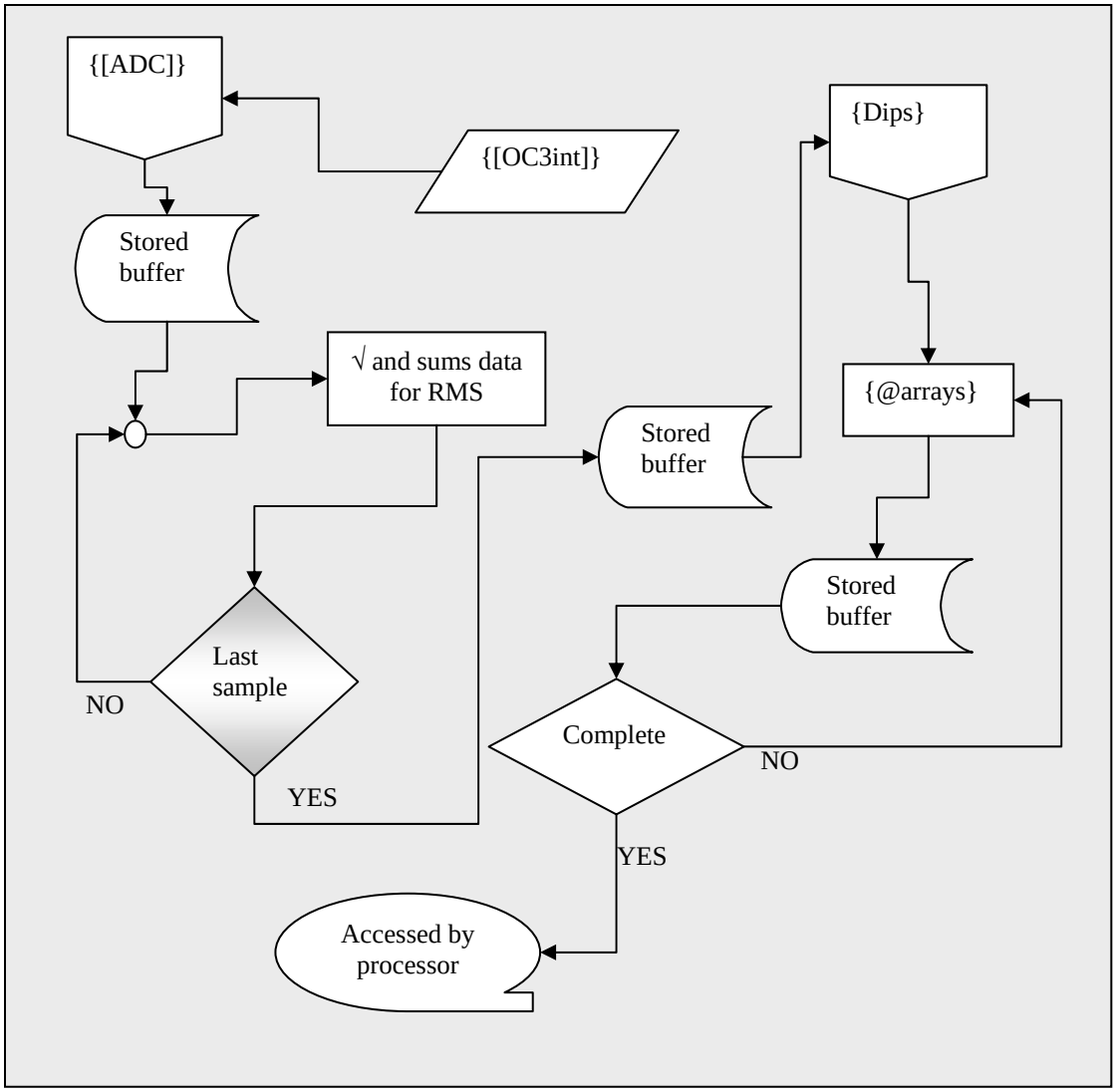


Fig. 4.13. Flow diagram of the sample task on the CME12-A4 single phase prototype.

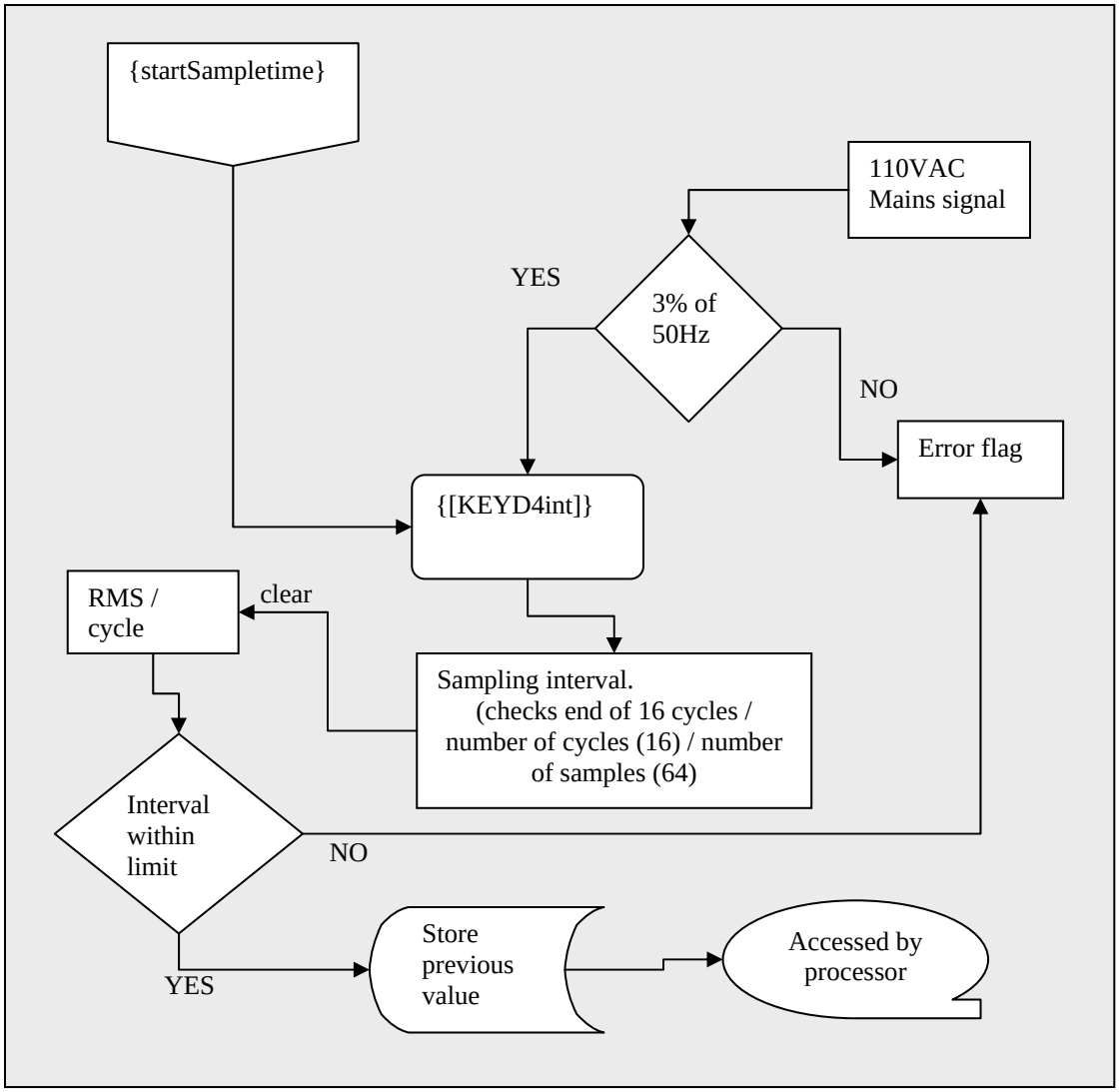


Fig. 4.14. Flow diagram of the microcontroller hardware interrupt environment to start cycle measurement on the CME12-A4 single phase prototype.

This concludes the hardware and software development on the single phase CME12-A4 prototype IED card interfacing to the RTU. The next part describes the steps followed during the period of interfacing the IED to the RTU via MODBUS. This covers the software configuration and techniques performed during the interfacing period of the single phase CME12-A4 prototype, as well as the final three phase product, using the MC68HC81284CPV8 microcontroller.

.

4.12. The MODBUS Interface from the IED to the RTU QUICC

The MODBUS protocol was discussed in chapter 3 [Ch.3.5, p. 58], and is used to allow communication from the single phase CME12-A4 prototype IED to the RTU QUICC card. This section explains the MODBUS coding.

4.12.1. Data coding for harmonics

A suitable MODBUS function for transferring 16-bit analog data is the Read Holding Registers function [Ch 3, p. 63]. The data available is 13 harmonics, THD, RMS and peak value, as well as error flags, a total of 16 words. A message block of 20 words have been chosen to allow for spare capacity if needed in future. The harmonics are calculated every 10 seconds; the data returned will be the latest available. Only certain Read Holding Register requests are accepted:

- Register 1-13 for the 13 harmonics,
- Register 14 for the Total Harmonic Distortion,
- Register 15 for R.M.S. value,
- Register 16 for Peak value,
- Register 20 for Error flags.

The data is presented such that 1000 represents 100.0%

Error flags: Bit 1 indicates that the frequency was out of limits (+-3%)

Bit 2 indicates loss of mains signal.

4.12.2. Data coding for dips and surges via MODBUS

A suitable MODBUS function for transferring bit data is the Read Input Registers function [Ch.3, p. 63]. See table 6.7 on page 185, showing the section error box for voltage dips and surges [Fig. 4.3, p.73], where the bit coding is:

- bit 1 NER code S
- bit 2 NER code T
- bit 3 NER code X
- bit 4 NER code Y
- bit 5 NER code Z
- bit 6 dips/surge event longer than the error box allows for
- bit 7 frequency was out of limits (+-3%)
- bit 8 loss of mains signal

The dips/surges are calculated every cycle (50 times / sec.), and an event is stored in an output buffer where it will remain until read by a MODBUS request [Ch. 3.5.2, pp. 59-61]. The output buffer is cleared after having been read and subsequent reads will show all zeros, or no event.

4.13. RTU Configurations

To test the final MC68HC812A4CPV board on the RTU, the three most important programs that were used were:

- The Universal Network Configurator (UNICON).

- The FieldComm program.
- DOS Program (VT Terminal).

4.13.1. MODBUS Plant settings on UNICON (Universal Network Configurator)

The Universal Network Configurator (UNICON) provides a user interface for the Portable Engineering Human Machine Interface (PEHMI) facility configuration engine. It presents configuration information in a graphical intuitive way enabling the user to interact with the stored facility configuration database. To accomplish the configuration on the RTU QUICC card, the following configurations were performed on the UNICON program for the analogue, as well as the digital input data used for this research:

- Read 16 bit Holding Register to AI - Offset 0 - No. **60**
- Read 16 bit Input Status to DI - Offset 0 - No. **6**

Fig. 4.15 shows the UNICON plant settings for the MODBUS on the IED device.

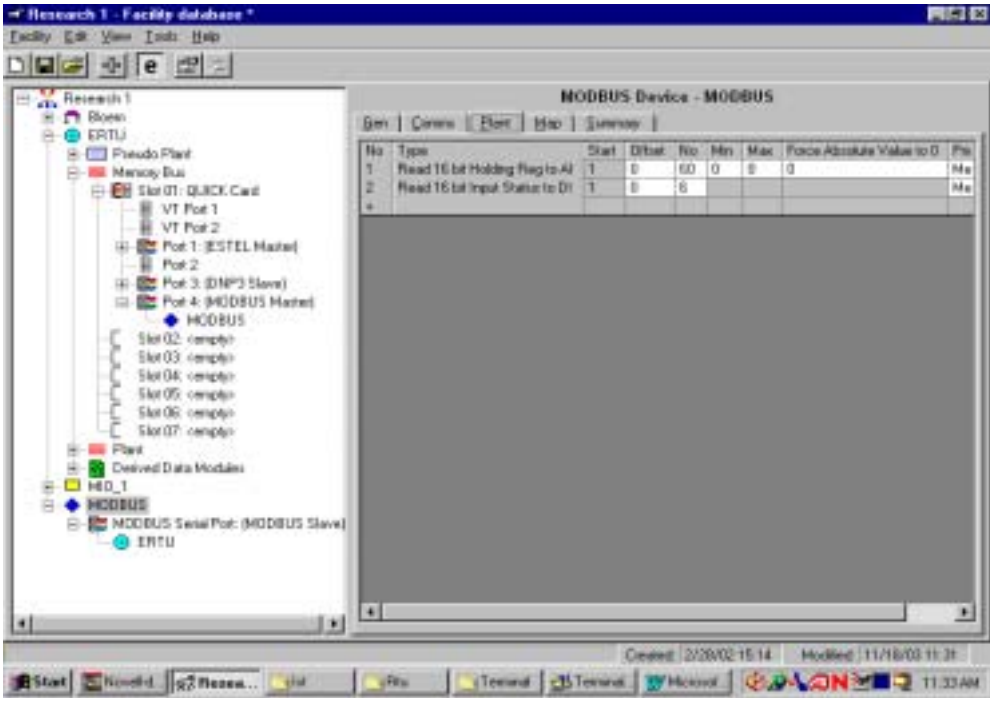


Fig. 4.15. UNICON Plant configurations for the MODBUS IED device setup.

4.13.2. MODBUS communications parameters

The same criterion applies for the communication parameters of the MODBUS protocol [Ch.3, pp. 59-61]. For the purpose of the evaluation tests, the IED device MODBUS address "**1"** was used, with exactly the same settings:

- Rx buffer size **248**
- IC Timeout (ms) **40**
- Retry Timeout (x100ms) **10**
- Retries **3**
- Baud **9600**
- Bits **8**

- Parity **None**
- StopBits **1**
- Modem Control **off**

4.13.3. Interfacing the IED to the QUICC Controller

The MODBUS, ESTEL and DNP-03 settings, as well as all Video Terminal on-line monitoring were all performed on the QUICC card, positioned at slot 1 of the RTU whilst testing the IED. On this card, the following criteria had to be applied before any configurations could be performed:

- Only one QUICC Controller card can be plugged into an RTU motherboard.
- The QUICC Controller is always located in position Memory 1 of the Memory Bus Area of the motherboard.
- The functionality of the QUICC Controller includes data processing for input/output subsystems, and the QUICC Controller firmware must therefore be compatible with Serial I/O Controller firmware.
- The QUICC Controller firmware version and the firmware version of all Serial I/O Controllers in the RTU must be the same (for research purposes version 2.7.11 was used).

4.14. Video Terminal (VT) program

The monitoring and diagnostic software configurations that were performed on the Research RTU were accessed by using a Personal Computer (PC) Laptop running VT emulation software. The VT was connected to the QUICC Controller, allowing the database of each to be examined or diagnostics to be run on the card to which the VT was attached. The monitoring and diagnostic software were used to perform two distinct functions:

- On-line monitoring of data.
- Running diagnostics with the ERTU off-line.

4.14.1. On-line monitoring of data

Selecting this mode of operation allowed the author to view the condition of all inputs and outputs associated with the RTU (if connected to the QUICC Controller). The information shown may be either unprocessed data or data subsequent to processing, according to user selection.

4.14.2. Running diagnostics with the ERTU off-line

This mode of operation allowed checking the internal circuit functions, serial port operation etc.

4.15. Connecting the Video Terminal (VT)

The connections for the VT were accomplished via RS232 from the laptop serial port to the No.1 or 2 connectors of the QUICC. Either No 1 or 2 ports can be used for the VT inputs, where one port can be used as VT input and the other for diagnostics, as shown figure 4.16.

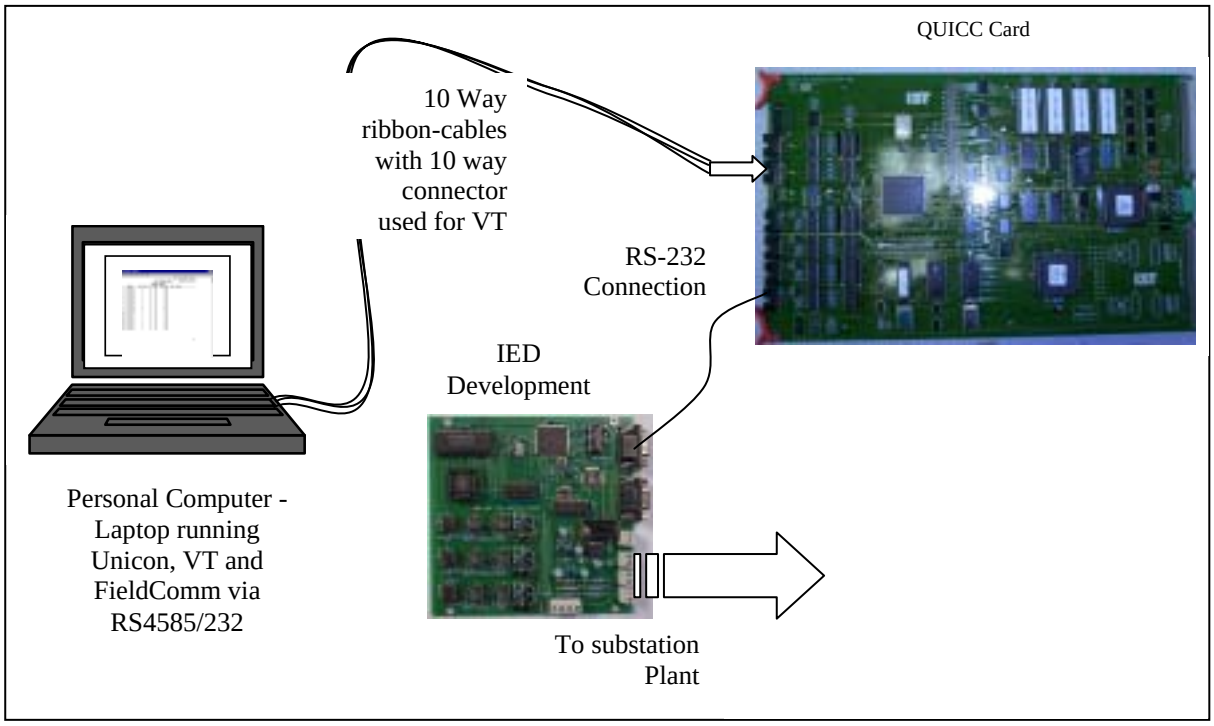


Fig. 4.16. Video Terminal connections to the QUICC card as used on the IED.

4.16. Video Terminal communications settings

The setting applied for the Video Terminal during this stage of the research was as follows:

- Transmit=9600; Receive = Transmit
- XOFF at 64; 8 bits, no parity; 1 stop bit
- No local echo

- RS232, Data leads only
- Disconnect 2s. delay
- Unlimited transmit; No Auto Answerback

4.17. QUICC controller on-line monitoring mode

With the VT connected to the QUICC Controller, selecting the Online Monitoring Mode, the current status of the QUICC Controller memory, database and ports, as determined by on-line background checks, was then constructed. Note that the QUICC Card was configured for the DNP-03 protocol to act as a Slave from the ENMAC Master, as a Master to the Multi Input/Output (MTIO) talking ESTEL protocol, and a Master talking MODBUS to the IED Card. The reader can view figure 6.18 for an overview of all protocol configurations used and shown below:

- QUICC to ENMAC – DNP-03 Protocol.
- QUICC to MTIO – ESTEL Protocol.
- QUICC to IED – MODBUS Protocol.

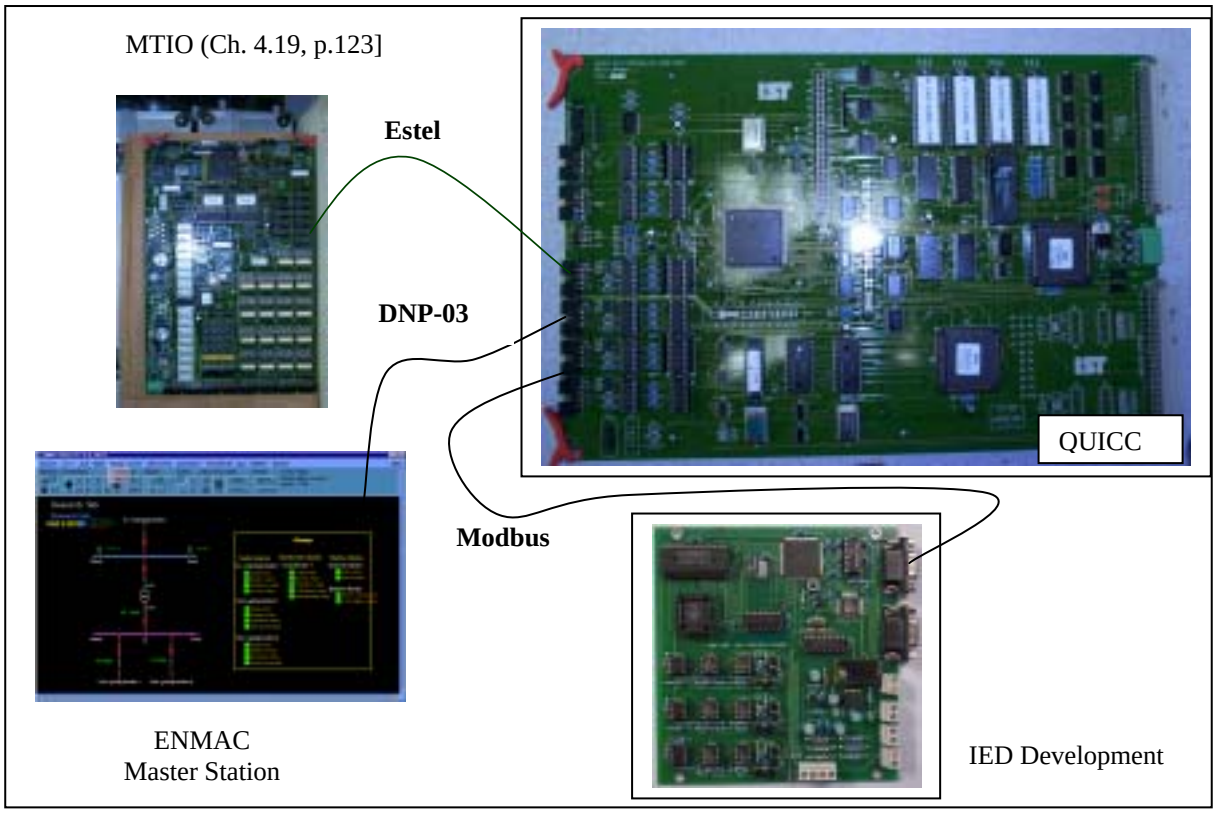


Fig. 4.17. Synopsis of the protocols used for the implementation of the research project.

The UNICON configurations in figure 4.18 also show the configuration of port 1 for the ESTEL Slave protocol that was used between the MTIO and the QUICC controller card, and port 4 for the MODBUS protocol between the QUICC controller and the IED development.

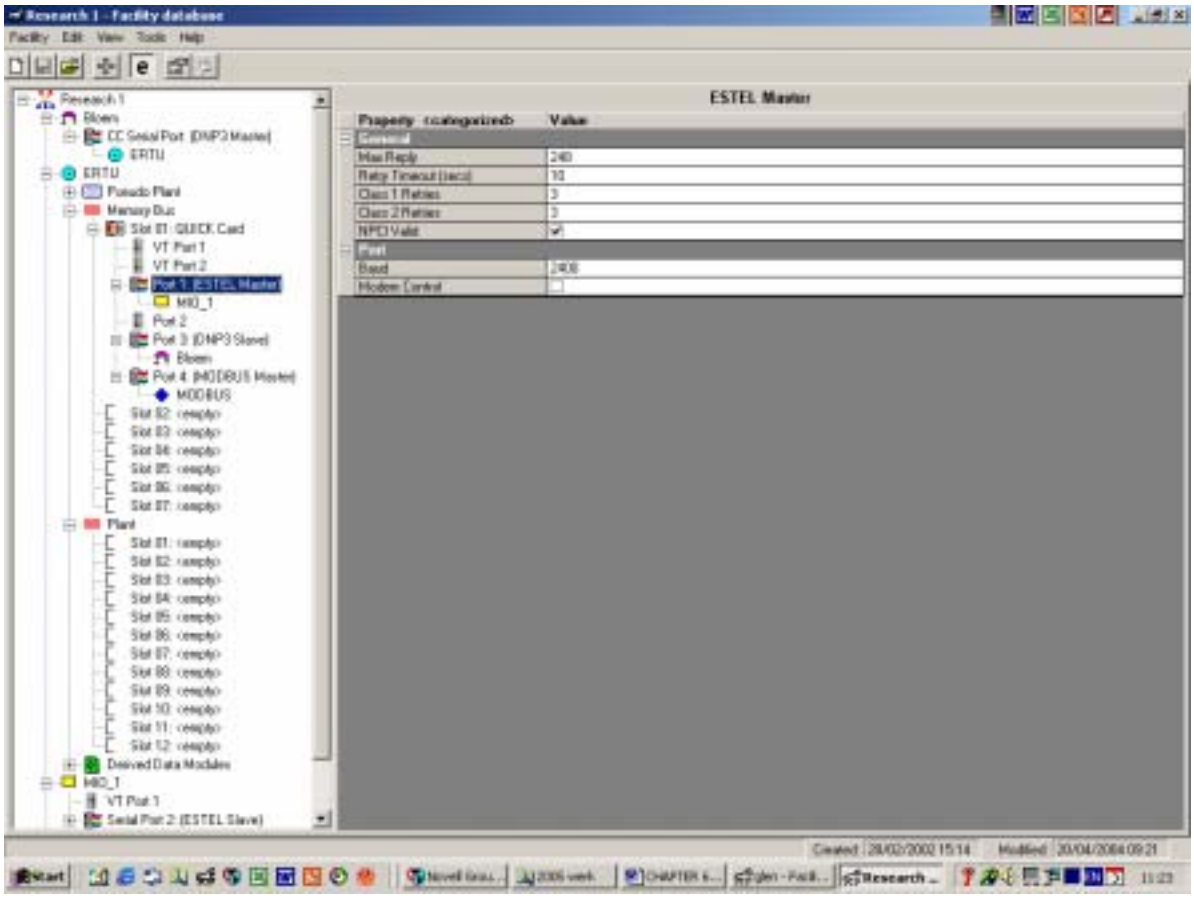


Fig. 4.18. UNICON port configuration for the ESTEL slave and MODBUS protocol.

4.18. MULTI I/O controller configuration

The MTIO shown in figure 4.17 (used to indicate the IED 5VDC and 12VDC power failure alarms of the IED, and also the substation general alarms), is a processor controlled input/output controller with:

- 64 digital inputs.
- 16 analogue inputs and
- 16 single secure control outputs.

It can be helpful to be aware that the MTIO can be used as a stand-alone RTU or slaved to a DRTU or bay processor. During this Research, the MTIO was used for the IED power indications, as well as primary Substation alarms like Breaker indications, Plant alarms and Substation analogues, as explained in the IED Substation hierarchy [Ch.5, p.164]. The QUICC card can now act as a master and will poll the MTIO as well as the IED to retrieve data.

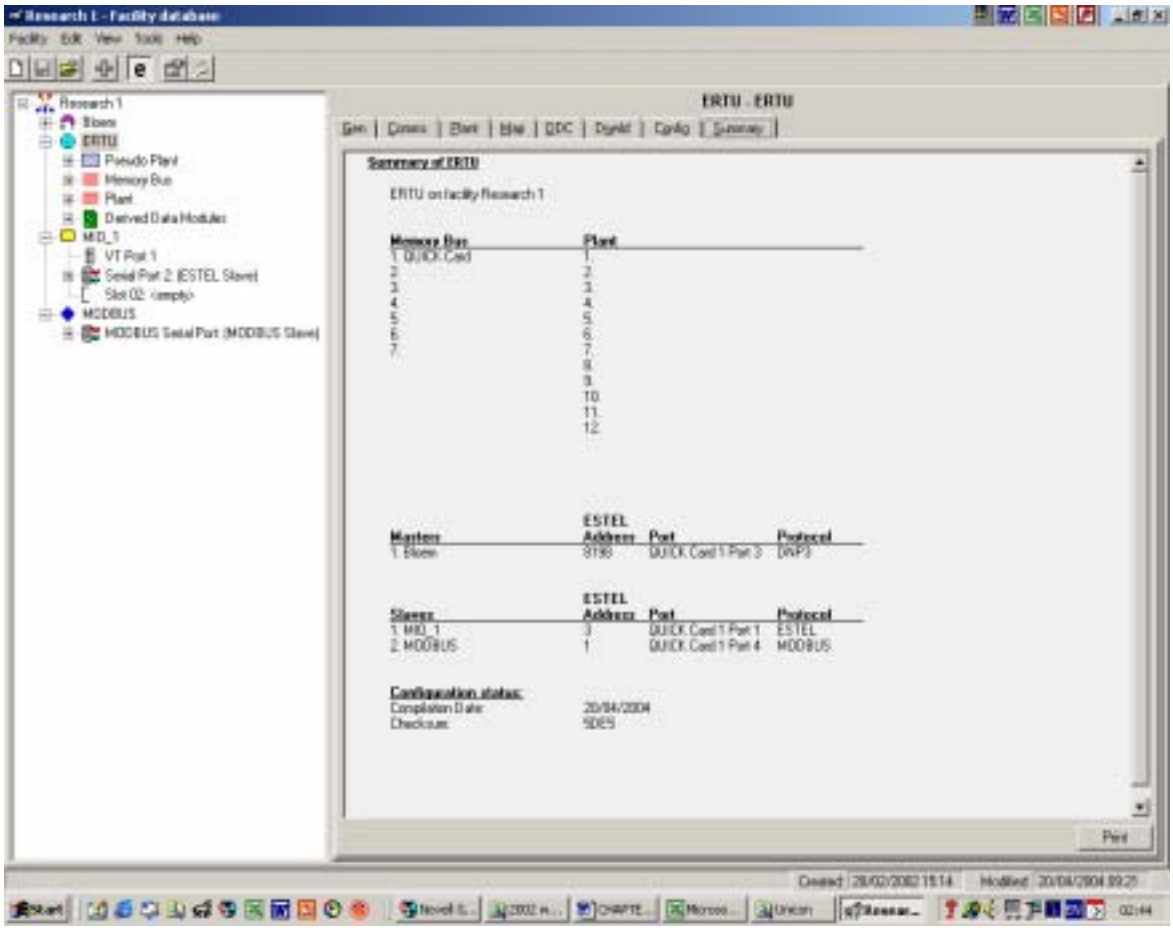


Fig. 4.19. UNICON display of QUICC card Master and MODBUS slave configuration.

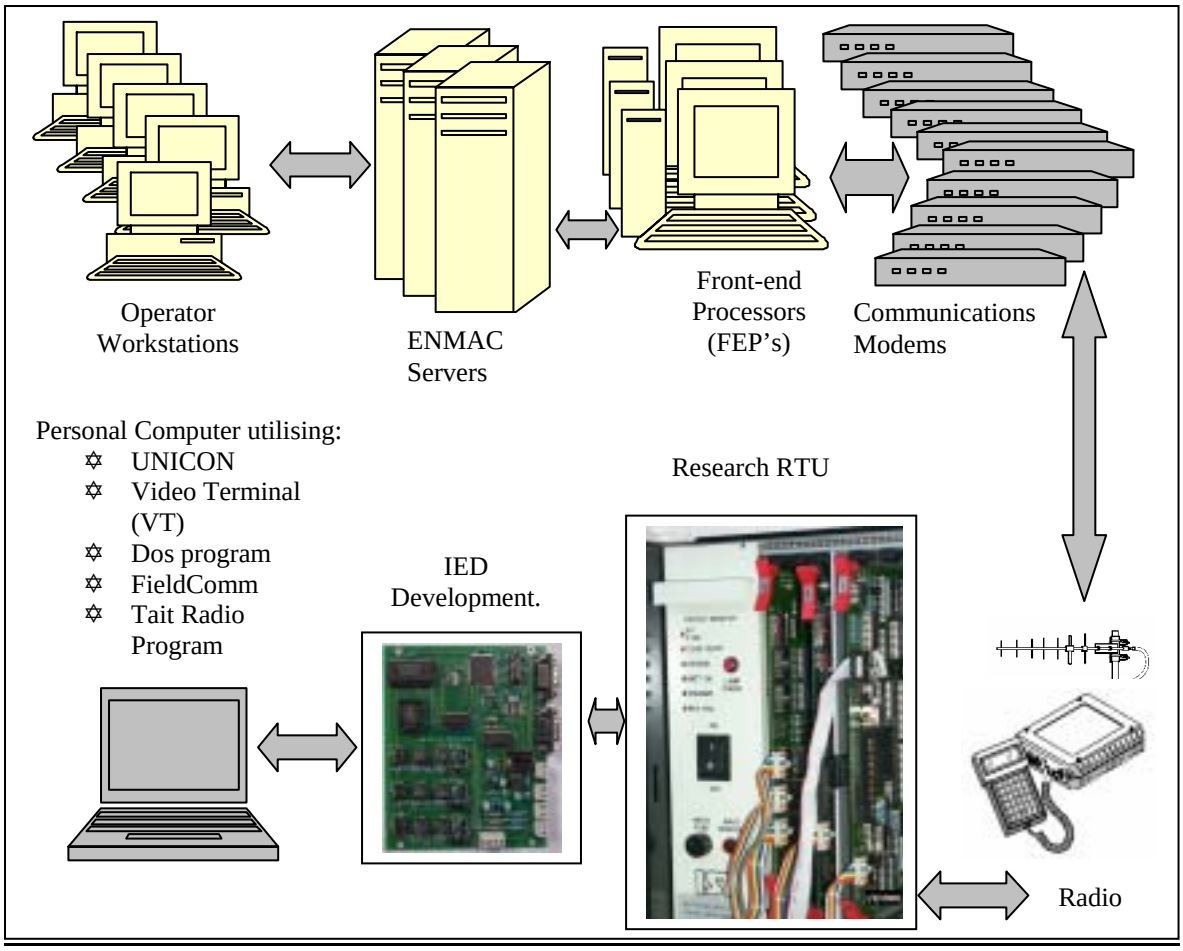


Fig. 4.20. Final hardware configuration setup as used for the evaluation period.

4.19. Digital and analogue parameters

In the same way as explained in this chapter [Ch.4.12.1, p. 112], the IED was designed for transferring 16-bytes of analogue data:

- 1-13 bytes for harmonics,
- 1 byte for THD,
- 1 byte for RMS value,
- 1 byte for error flags,

One MULTI I/O allows for 32 bytes of analogue inputs. For this research only the first 16 bytes were allocated for the normal substation MULTI I/O analogues, like substation bus-bar voltage, ampére, MW and MVar. The MODBUS plant configuration for the IED was configured to allow the next 60 x 16 bit holding registers to the analogue inputs (therefore from 17 -76), and for 6 x 16 bit digital input states. Therefore the mapping of the MTIO and MODBUS IED device were done as follows:

Table 4.8. Analogue database mapping table for MTIO and MODBUS.

Database number	Source	Tag	Address
1-16	MTIO	MTIO_Estel	D3_AI_NO 1-16
17-76	MODBUS	MODBUS device	D4_AI_Grp 1_No1-76

The MULTI I/IO also allows for 64 physical digital input points, thus 64 bits, and for the IED it was configured to be used from 65 digital inputs onwards, as shown in table 4.9.

Table 4.9. Digital input database mapping table for MTIO and MODBUS.

Database number	Source	Tag
1-64	MTIO	D3_DI_Grp 1-7 (8 groups) of B#1-8 (8 groups) total of 8X8 = 64).
65 - 112	MODBUS	D4_DI_Grp 2_No1-Bit 1-8 up to No 6-Bit 8.

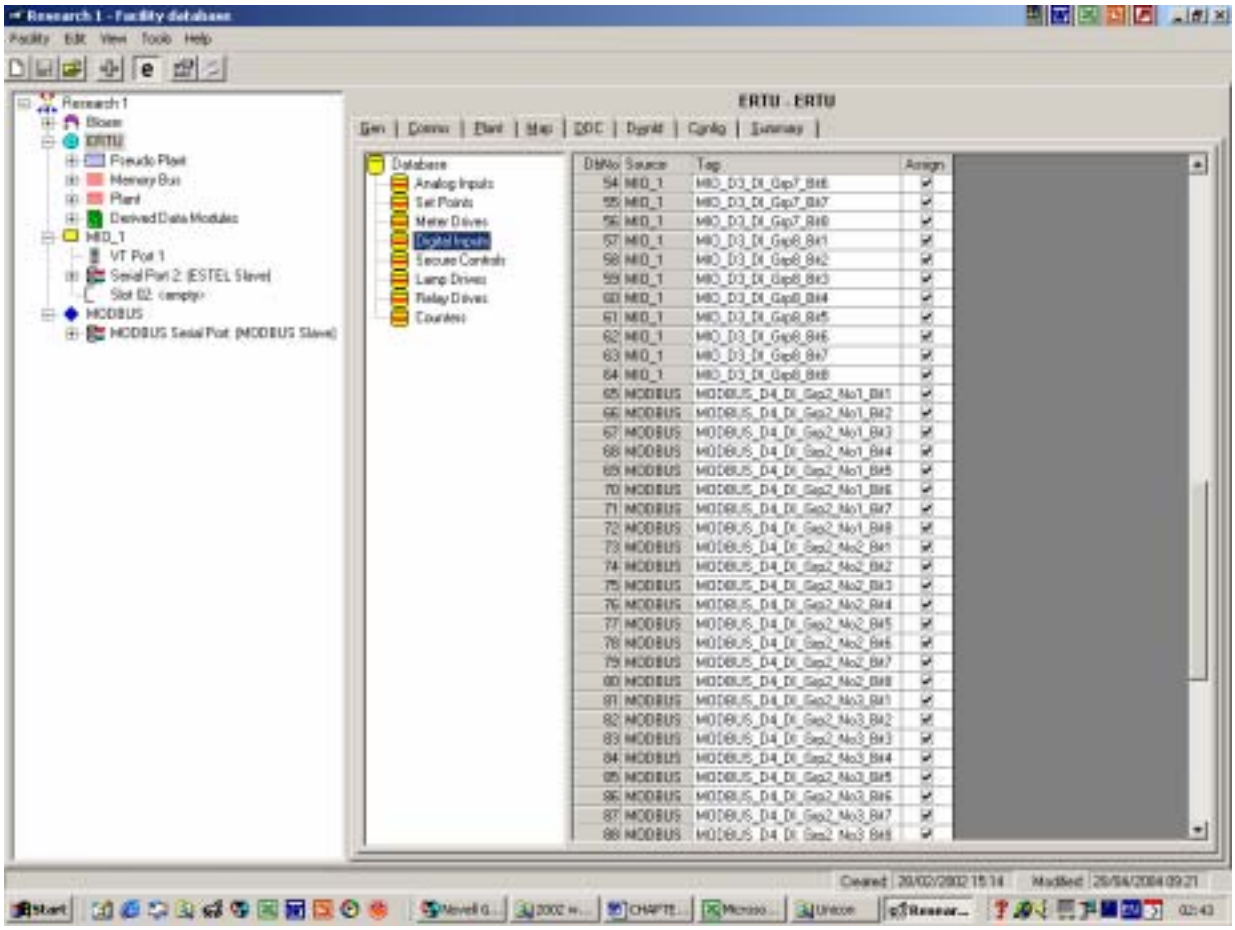


Fig.4. 21. UNICON digital input mapping configuration for the MTIO and MODBUS device.

4.20. Dip configurations at ENMAC

At the ENMAC side, the MODBUS IED device acted as a slave, talking MODBUS to the RTU, and the RTU were talking DNP-03 to the ENMAC Master station. For the dip indications the ENMAC was configured with 8 digital inputs (DbNo. 57-64); Digital input 1-8 as shown in figure 4.21:

- Digital input 0: S – Dip (dBNo. 65)
- Digital input 1: T – Dip (dBNo. 66)
- Digital input 2: X – Dip (dBNo. 67)

- Digital input 3: Y – Dip (dBNo. 68)
- Digital input 4: Z – Dip (dBNo. 69)
- Digital input 5: Error state – long dip (e.g. breaker tripped or brownout) (dBNo. 70)
- Digital input 6: Frequency out of limit (dBNo. 71)
- Digital input 7: Loss of mains signal (dBNo. 72)

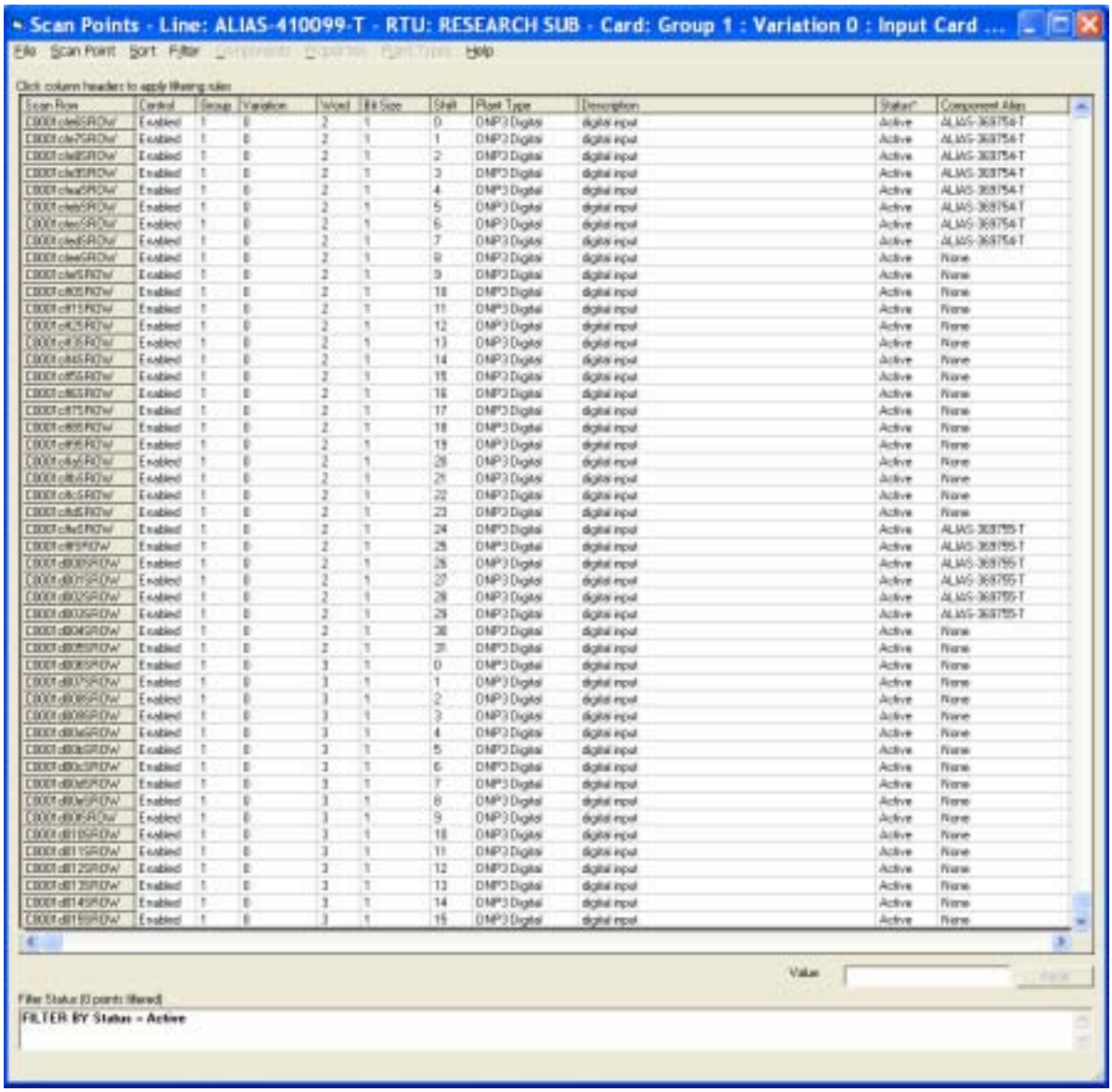


Fig. 4.22. ENMAC digital input configuration for the IED dip indications.

4.21. Tait T2015 UHF radio configurations

The communications software programming was done using the T2000 conventional programming application shown in figure 4.23. For the purpose of the research tests, a UHF frequency of 406,1250MHz was used in direct mode to ENMAC for this research. Although the Tait T2015 radio with internal FFSK was not used prior to this research, the programming issues will not be reviewed as it is not really relevant to the research.

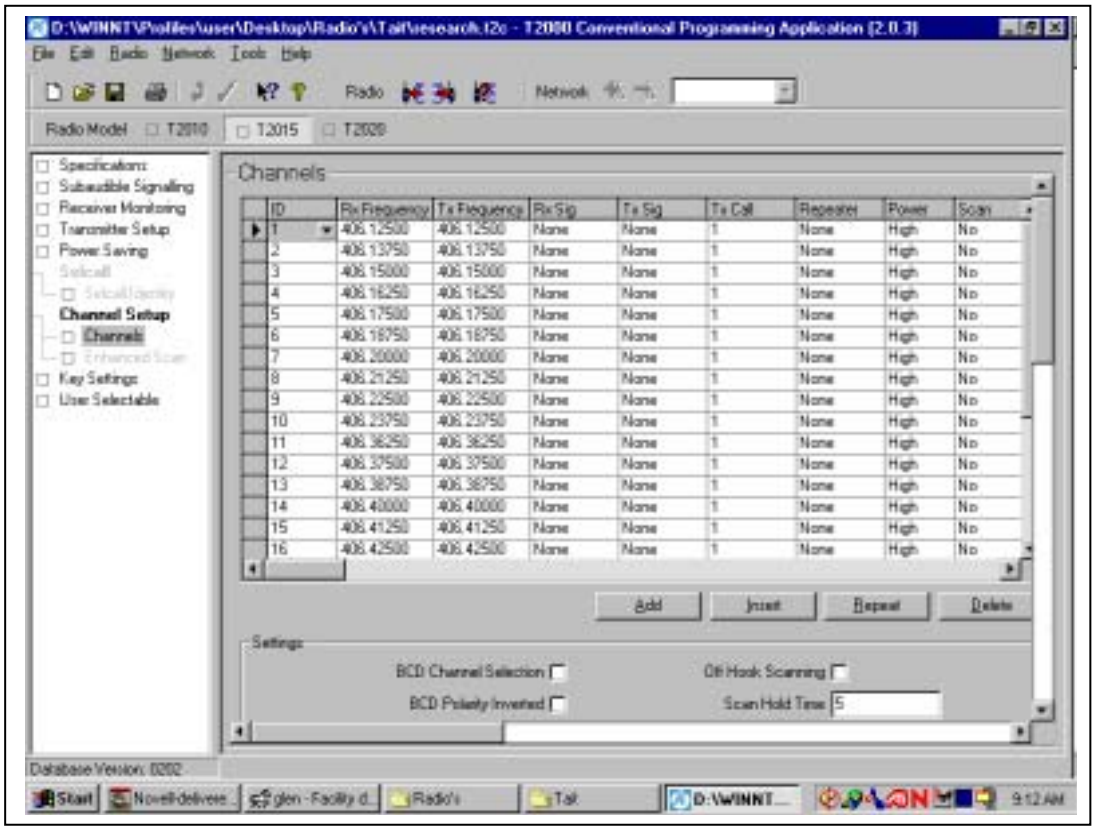


Fig 4.23. Radio configurations as used for this research.

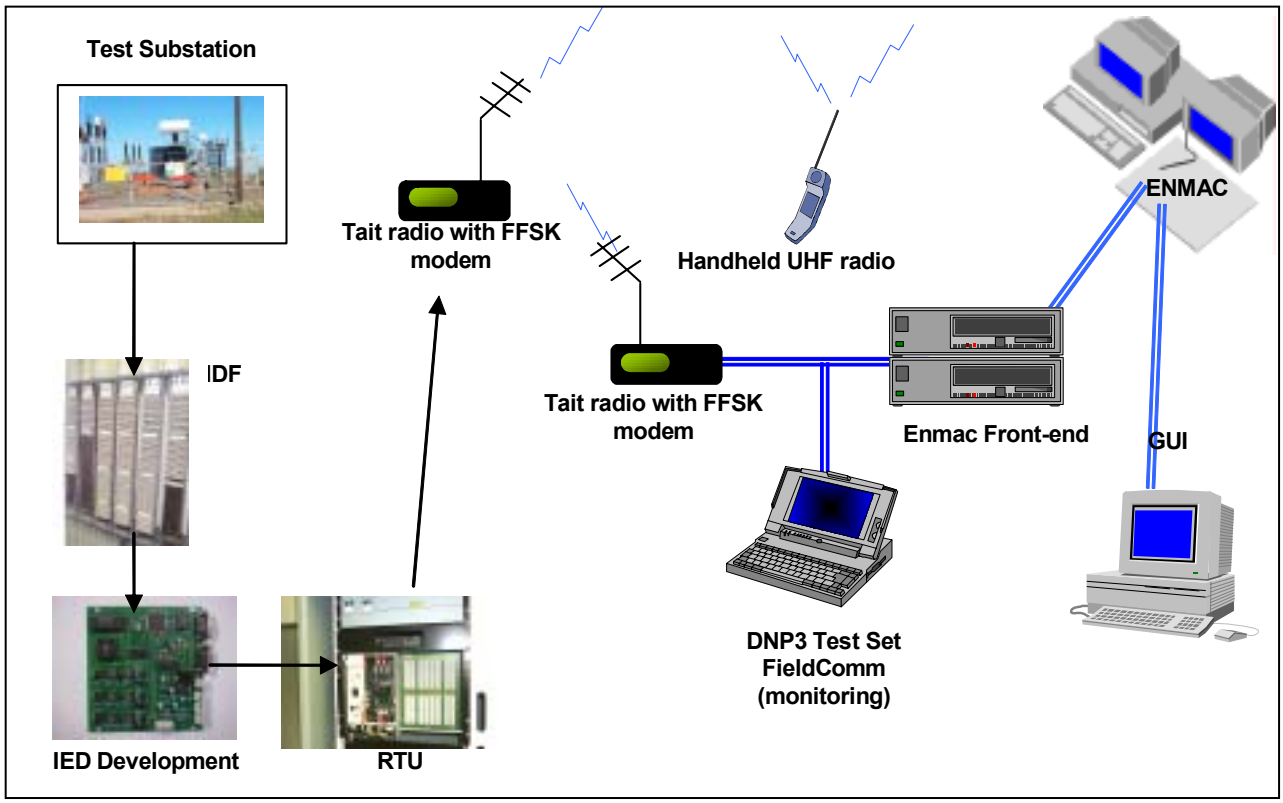


Fig. 4.24. Diagram showing the final configuration incorporating the SCADA and IED that was used at Glen substation.

4.22. Summary

During this chapter, the parameters outlined in the project plan were configured, and can be summarized as follows:

- Development of the input circuitry of the measurement unit to the Substation plant.
- Implement the measurement circuitry, initially the ADSP-2181 EZ-KIT Lite and software.
- Integrated the developed input circuitry to the ADSP-2181 EZ-KIT Lite and software.

- To prove measurement reliability, a test collection test program was then developed, using LINUX, to generate waves that simulate real harmonics and voltage dips. This development allowed for the mathematical generation of a sine wave with 13 harmonics of different amplitudes, which could be used for testing.
- Modification of the above collection to simulate dips as outlined in the NER specifications.
- Implemented the algorithm from the ADSP-2181 EZ-KIT Lite onto the PC running under LINUX and test it with Simulation collection 1.
- Extend the code to include Total Harmonic Distortion.
- Extend the code to detect voltage dips.
- Test the new collection with Simulation collection 2.
- The Axiom CME12-A4 development unit was then implemented, using single phase only, to test with the developed LINUX test program.
- Integrated this to the RTU and the developed input circuitry, and the prototype was tested adequately according to the NRS specifications.
- This was then configured and tested to the ENMAC master station.
- The final product was then assembled, using the MC68HC812A4CPV microcontroller.

The parameters outlined for the development were implemented to simulations developed in this section. The steps are outlined as follow:

- Produce a suitable hardware platform for the IED (Intelligent Electronic Device) on a bread-board prototype.
- Implement harmonic detection software on the hardware developed. If the hardware performed has sufficient processing power, monitor three phases, if not; monitor only one phase to prove the concept.
- Extend the software categories on the dips as outlined in the NER specifications (for the harmonic part of this development, this was prepared only up to the 13^{th} harmonic, as explained in 4.1).
- Implement a specific protocol (MODBUS) on the IED to allow the hardware to communicate its data with the external device, the RTU QUICC card.

Also during this section it was found that the processor used did have sufficient processing power to implement 3 phases, so the code was written accordingly. At the development stage:

i) The actual bread boarded prototype was only designed for one phase to minimize hardware complexity and cost - but;

ii) The final board layout was designed for all three phases.

On the single phase bread boarded prototype, all three the algorithms developed, harmonic detection, THD analysis and dip detection were ready to be tested in a simulated environment as explained in chapter 6.9, to be proved satisfactory for the stipulated requirements.

It was also during this period that the RTU QUICC card was identified as the preferred external device to interface to the RTU.

The IED design was for transferring 16-bytes of analogue data, e.g. 1-13 bytes for Harmonics, 1 byte for THD, 1 byte for RMS value and 1 byte for error flags. One MULTI I/O allows for 32 bytes of analogue inputs, of which only the first 16 bytes were allocated for the normal substation MULTI I/O analogues, like substation bus-bar voltage, ampére, MW and MVar.

The MODBUS plant configuration for the IED was configured to allow the next 60 x 16 bit holding registers to the analogue inputs, and for 6 x 16 bit digital input states.

At the ENMAC side, the MODBUS IED device acted as a slave, talking MODBUS to the RTU, and the RTU were talking DNP-03 to the ENMAC Master station.

CHAPTER 5

ENMAC HARDWARE AND SOFTWARE CONFIGURATIONS

5.1. Introduction

To become familiarised with the outcome accomplished during this research project, a very brief discussion about the end-user equipment and software, called the Electricity Network Management and Control (ENMAC) system follows.

The ENMAC SCADA system that was used for this research project is linked to various remote substations, and reports to a Master station, which is situated at two separate centres. This system controls and monitors substation plant such as breakers, transformers, capacitor banks and bus couplers, and consists of different RTU's, using unique protocols like ESTEL, PUTU, INTRAC and the latest international recognised protocol, DNP-03, which was pioneered in the ESKOM environment with great complexity for the successful implementation of this research project.

5.2. The ENMAC hardware

The Eskom North Western Region SCADA system is located in two control centres. It is based upon Digital AlphaServer and Compaq client workstation configurations, using Windows NT and Digital UNIX server software. Both control centres run dual Local Area Networks (LAN's) that are routed to the Eskom corporate Wide Area Network

(WAN) and to the Eskom 2 Mega bits per second (Mbs) Real Time Data Network (RTDN) respectively. A 2 Mb/s link allows both the control centres a primary link to communicate with their respective local Front End Processor's (FEP's), which are used for communications with the RTU's via different modems. The corporate Wide Area Network (WAN) connects the two control centres in such a manner that the databases in the Servers in both centres are kept identical at all times such that if needed, either control centre can take over the activities of the other.

To be able to visualise the RTU's, several triple-headed, two-headed and one-headed workstation clients are connected to a LAN. These workstations run the ENMAC software, (which will not be discussed in this thesis), and are also configured as NT clients. The workstations are equipped with two operational Digital UNIX Symmetrical Multi-processor serveReplication Servers (SMP's), running the ENMAC Data Management Servers (DMS) and ENMAC SCADA software.

There are two Front End Processors (FEP's), of which the one which is housed in Bloemfontein has 48 dual dial-up Public Service Telephone Network (PSTN) serial lines, and the one in Kimberley has 24, which are connected to modems (fig. 5.1). Each server contains an accurate identical copy of the databases providing a fully fault-tolerant distributed database, using asynchronous symmetrical data replication. The ENMAC main server shares the load, but in the unlikely event of a failure of one of the servers, the other Server assumes total responsibility automatically without any operator intervention.

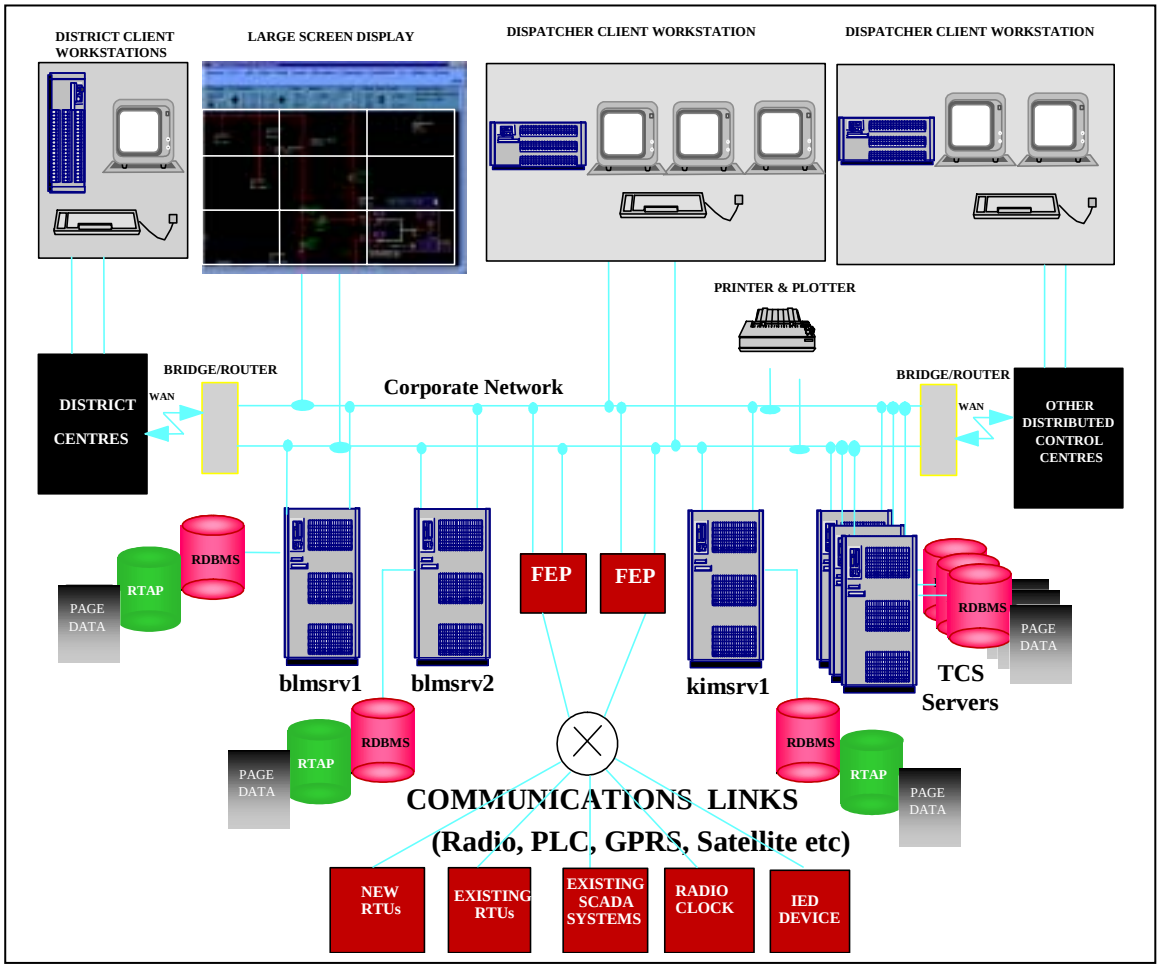


Fig. 5.1. The Front End Processor (FEP) functional overview.

The ENMAC system uses Real Time Application Process (RTAP's) plant database, scan system and calculation engine (CE) to provide SCADA functionality. The calculation engine (CE) enables mathematical functions to be applied to scanned values. For example, the scan system may return raw values that need to be scaled for engineering units; this is achieved using a calculation engine function.

The ENMAC system has more than one physical server to provide a reliable distributed system which provides for a great deal of flexibility and redundancy. The Network

Management Server (NMS)/SCADA system is made up of the pairing of logical NMS and SCADA servers. The paired NMS and SCADA servers reside on the same piece of hardware (blmsrv1 and 2, and kimsrv1), however they still run as logically separate servers.

The plant database holds the telemeter plant items, while the scan system handles the communication lines and the RTU's used to drive the scanning and telecontrols of the remote devices. The calculation system allows mathematical functions to be applied to the scanned values. The four main physical components are [18, p. 2]:

1) **The Client** – This is the computer workstation from where the user views the telemeter items and performs telecontrols.
2) **The NMS** - The NMS generates alarms that it will forward to all clients. It also passes telecontrols requests down to the SCADA server.
3) **The SCADA server** – It carries the RTDB and RTAP.
4) **The Front End Processor (FEP).** This is used to offload some of the processing away from the SCADA server onto a separate machine. The FEP performs all the polling of the remote devices and reports back to the SCADA server.

A diagram, showing the layout of ENMAC system overview as it was configured during the period of this research can be seen in figure 5.1. To process measurements of the Intelligent Electronic Device (IED) at the ENMAC's system overview, the three main types of interactions between the NMS and the SCADA server are [18, p. 2]:

1) **Transactions** (e.g. RTU communications and interrogation) are dealt with by the Transaction Manager (TMNGR) on the NMS and the SCADA Manager (SCAMNGR) on the SCADA server.
2) **Events** from RTAP (e.g. harmonic values, voltage dip and surge alarms, all other alarms and indications) are dealt with by the NMS event and SCADA event tasks.
3) **Telecontrols and SCADA commands** (e.g. telecontrols open) are dealt with by the NMS control and SCADA control tasks.

At the central station, the ENMAC Graphic User Interface (GUI) was used to provide access to the network entities, to be controlled in a user-friendly fashion. The phases utilised to complete this part of the research project are shown in figure. 5.2.

All the data that were generated by the IED development as explained during the course of Chapter 4 are stored at the three main data stores [18, p. 2]:

1) The **Real Time Database** (RTDB) which exists on the SCADA server. It holds all the data relevant to the scanning and control of telemeter items.
2) The **ORACLE database** exists on the NMS server. It holds all the static data, like analogues at which value a high alarm should be generated, and other information not relevant to SCADA (e.g. transformer No's, MVa's, etc.).
3) The **Graphic Real-Time Values** are kept at the NMS server. This holds the dynamic data associated with the telemeter items e.g. scanned values.

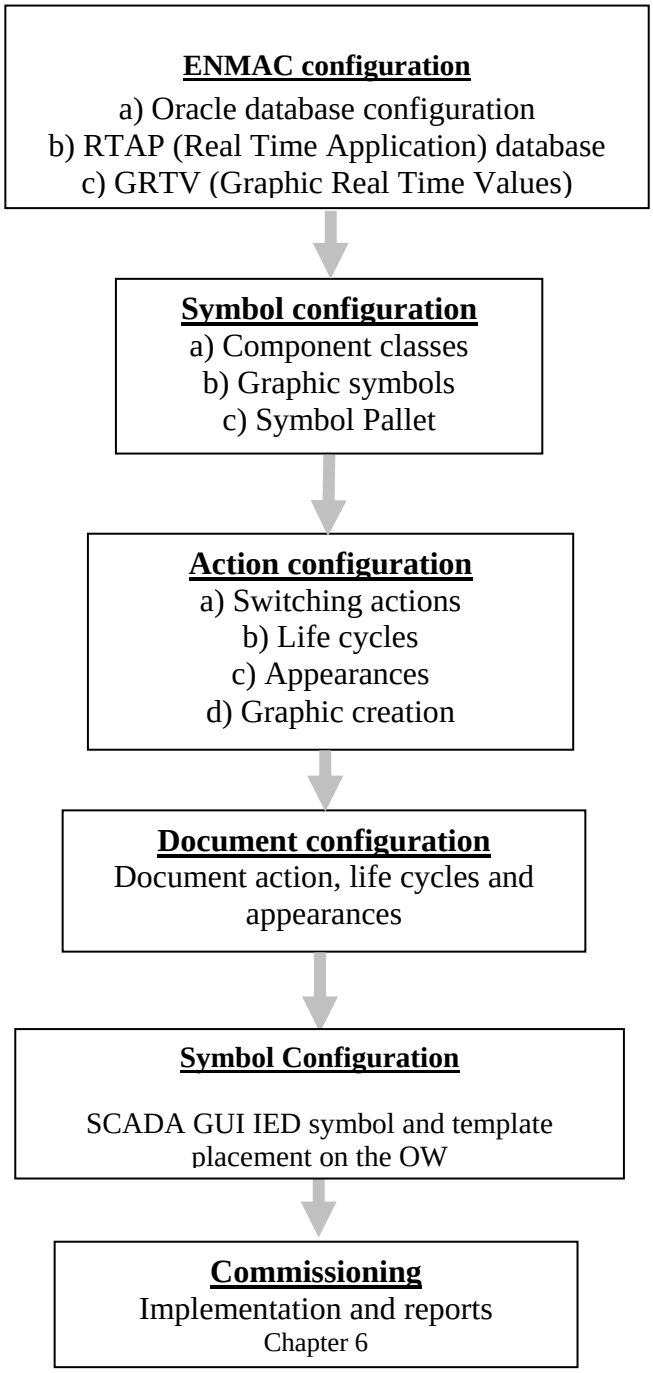


Fig. 5.2. Flow diagram showing the ENMAC configuration process during this research.

Scanned values are placed in a piece of memory called the Graphical Real Time Values (GRTV) that is a shared memory segment. When a client looks at a page with telemeter values, the GRTV on the server is queried and the relevant GRTV entries are fetched and displayed on the client's screen.

The scan system used for communication to the IED unit is composed of three parts, i.e.

- ☺ **Communication ports.** This part specified the communication protocol between the programs that do the scanning and the scan devices.
- ☺ **Scan devices.** This part defined information specific to each device, (e.g., RTU). The programs that do the scanning read this information and scan accordingly.
- ☺ **Scan tasks.** These are the programs that do the scanning. There is one scan task for each Communication Port.

Scan Tables are attributes of a scan device and they determine what is to be scanned and how it will be scanned.

5.2.1. The ENMAC Front End

At the Front End, the Real Time data received from the RTU is fed from the FEPs into the RTAP database. The ORACLE Relational Database Management System

(RDBMS) holds information about every component in the network. The server stores the network diagram pages and each client accesses only the information it requires.

5.2.2. ENMAC / RTU polling

The three main components that are responsible for intelligent polling of RTU's comprise of:

- ☺ The **Scan Task** (on the SCADA server).
- ☺ The **Distfep,** (on the SCADA server), and
- ☺ **Netfep** (on the Front End Processor).

The Front End Processor tasks can be started and stopped remotely or locally by the administrator. On initial start-up of each FEP (i.e. Scan Task), configuration information is downloaded to the Front End Processor to allow the formulisation of a scanning pattern, the Front End Processor tasks then report changes to the SCADA servers by exception (i.e. changes only).

For the goal of this research, the FEP was configured with the port number required for communication in the server database. A Transmission Control Protocol (TCP) port number was uniquely assigned to each scan task, which runs on the server. In figure 5.1 there are two RTU types configured under that FEP, where the research RTU was represented as the "New RTU's", with its physical addresses configured. In

addition, the research RTU was configured to be polled on the line whose properties are described in the line configuration table in the FEP point. This configuration includes properties such as baud rate, parity, bits per character, timeouts, flow control, etc. as necessary and also shown by the UNICON configurations in chapter 4 [Ch.4.13, pp. 113-117].

5.2.3. The Scan Task process

When the ENMAC operator issues an ENABLE command, one scan task process is started on each SCADA server. This scan task process sets up a TCP port ready for communication with a corresponding Front End Processor. The scan task will read all RTU (including the IED development) and communication data associated with this FEP from the database and will then wait for a connection to be established by the Front End Processor. Once a connection is established, the Master reads the total RTU database and all the configuration data is downloaded to the FEP, as shown by the FieldComm program [Fig. 6.12, pp. 196]. Scan data reported back by the FEP is written to the local database and replicated to all other active SCADA servers. Control requests and force poll requests are forwarded to the FEP as required.

5.2.4. The Supervisory process (NetFEP)

The ENMAC/FEP comprises of three main components which are responsible for intelligent polling of RTU's. These are the **Scan Task** (on the SCADA server) and the **NetFEP** and **DistFEP** (on the Front End Processor). Both the NetFEP and DistFEP

processes are started from the desktop (on MS Windows NT 4.x platforms), however to facilitate unattended reboots both of these processes were installed as kernel services. This approach ensures these processes are started without the need for user intervention (i.e. before a user **'login')**.

5.2.5. Multiple DistFEP setup

To effectively provide for the use of the FEP as an intelligent distributed data concentrator over a wide area network, the set-up of a separate FEP system was performed during the execution of this project by adding one DistFEP/NetFEP pair on one CUBIX NT Front End Processor. This configuration had the following advantages.

- It reduced the effect on the system of re-initialising a single FEP. A FEP may be disabled or enabled on the ENMAC diagram many times in the course of commissioning or in order to affect routine data changes. This was very helpful as during the initial tests of the IED and research RTU, various changes occurred, and in a radio system as used during this research, where RTU's cannot be quickly scanned and changes are predominantly reported by exception, this would minimise the re-initialisation of remote RTU's.
- It speeds up the initialisation of a single FEP. Obviously the initialisation time depends on the amount of data being downloaded at start-up. Therefore the less

RTU data associated with an FEP, the faster the data download will be. This factor became noticeable when one FEP was scanning many large RTU's.

5.3. ENMAC software

ENMAC uses the UNIX v4.0 operating system, as well as other applications like Real Time Application (RTAP) database, Oracle database and Windows NT. The Application Programming Interface (API) holds the NMS, RDBMS, RTDB & SCADA systems.

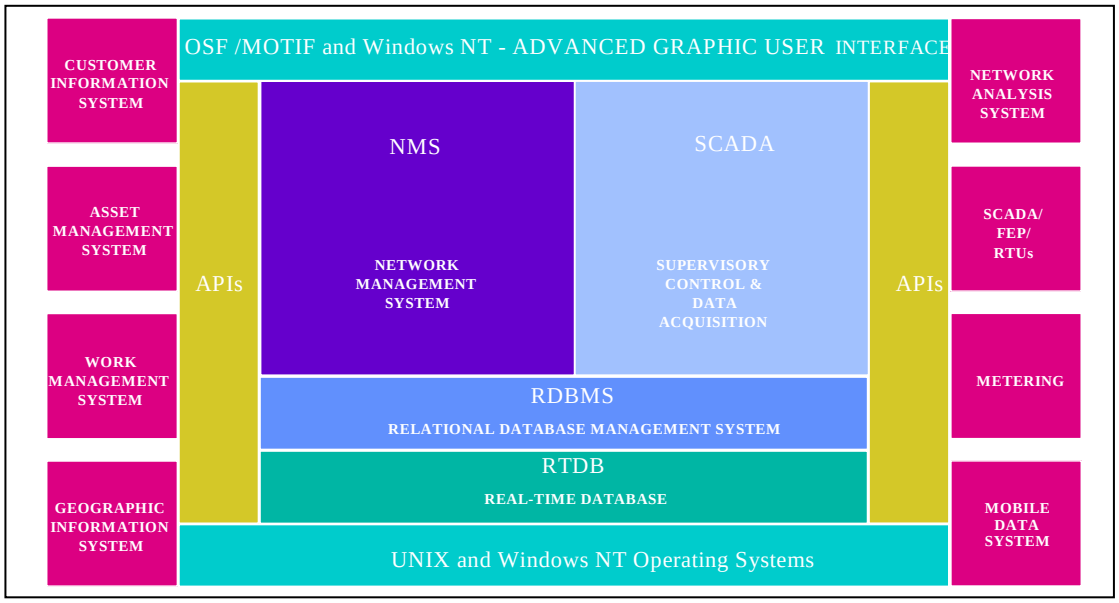


Fig. 5.3. Block-diagram showing the layout of the ENMAC system overview.

5.3.1. Real Time Application Process (RTAP)

RTAP, which was persistently used during this project, is an integrated family of SCADA software applications, which can be manipulated to meet the requirements of real-time data capture and analysis systems [18, pp. 3, 4].

It entails a set of core tools to provide configurable functionality for real-time control and data acquisition systems like managing, transferring and manipulation of data.

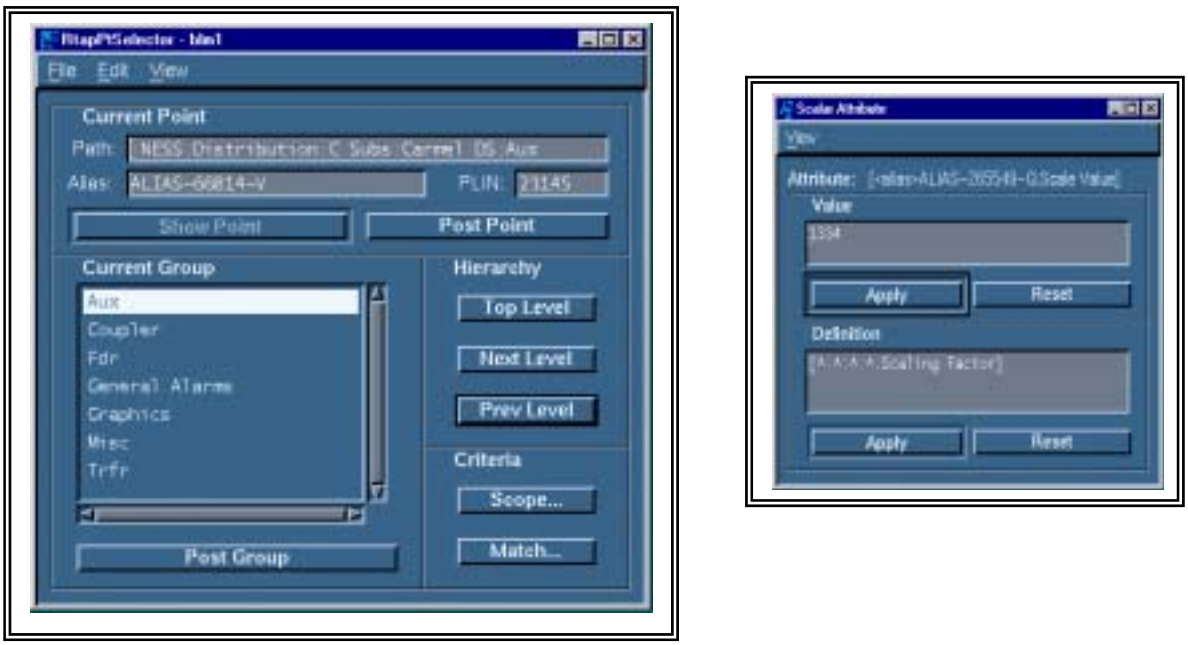


Fig. 5.4. Typical example of the RTAP tools as used during this research.

5.3.2. The Oracle database program

The Oracle database programme was extensively used to maintain the whole plant database hierarchy, while RTAP only contains a subset of the plant database hierarchy. Oracle forms are used to configure all aspects of ENMAC like alarms, telecontrol actions and life cycles, documents, symbols, and archiving. [19, pp. 1-4],

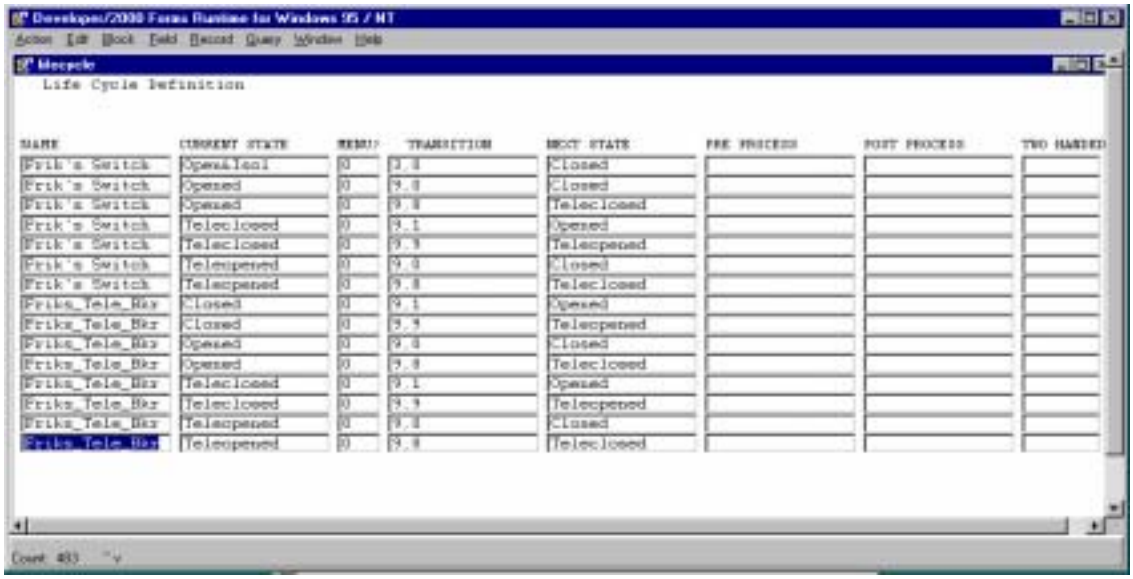


Fig. 5.5. View of the Life cycle definition table in Oracle.

5.3.3. The ENMAC operating system

ENMAC uses the UNIX v4.0 operating system. UNIX and its command line interface allow complex jobs to be rapidly executed with a relative short command [19, p. 2]. A shell program, in this case K shell, reads the commands, and processes them. The K shell is an interactive command interpreter – a program that translates the UNIX commands entered by the user to a low-level instruction, which the system can carry out.

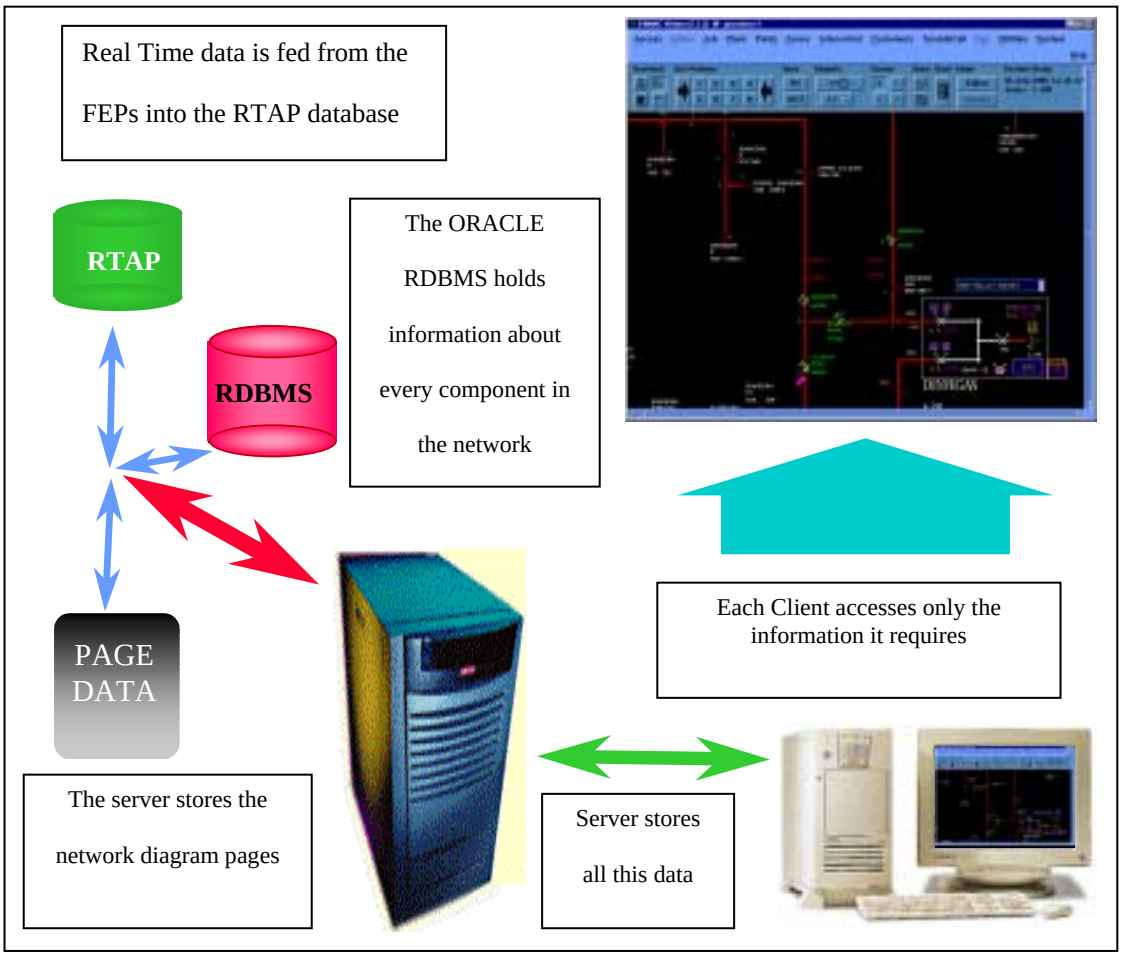


Fig.5.6. The outlay of the Front End Processor (FEP), showing the ORACLE RDBMS and RTAP [19, p. 5].

5.4. Research symbol configuration

All plant items on a client's screen consist of graphics and a database object that drives this graphic. For a given database object, there can be many graphical representations.

5.4.1. Dynamics

If the symbol is a SCADA symbol, then the state may be displayed by *dynamics*. A dynamic is a graphic primitive that is driven by an external source (e.g., if the colour of a line is driven by an RTAP calculation, it is known as a dynamic). RTAP provides the calculations to drive the dynamics.

5.4.2. The Graphic symbol

The graphic symbol consists of an outer box, (which was green for the purpose of this research), as well as the filled inner box, (which will by dynamically tied to an RTAP calculation that works out the colour depending on whether it's open or closed).

5.4.3. Switchgear Graphical symbol

The procedure for creation of the graphical symbol is shown in figure 5.7, using the Control Panel and the Editor Tools programmes in figure 5.8.

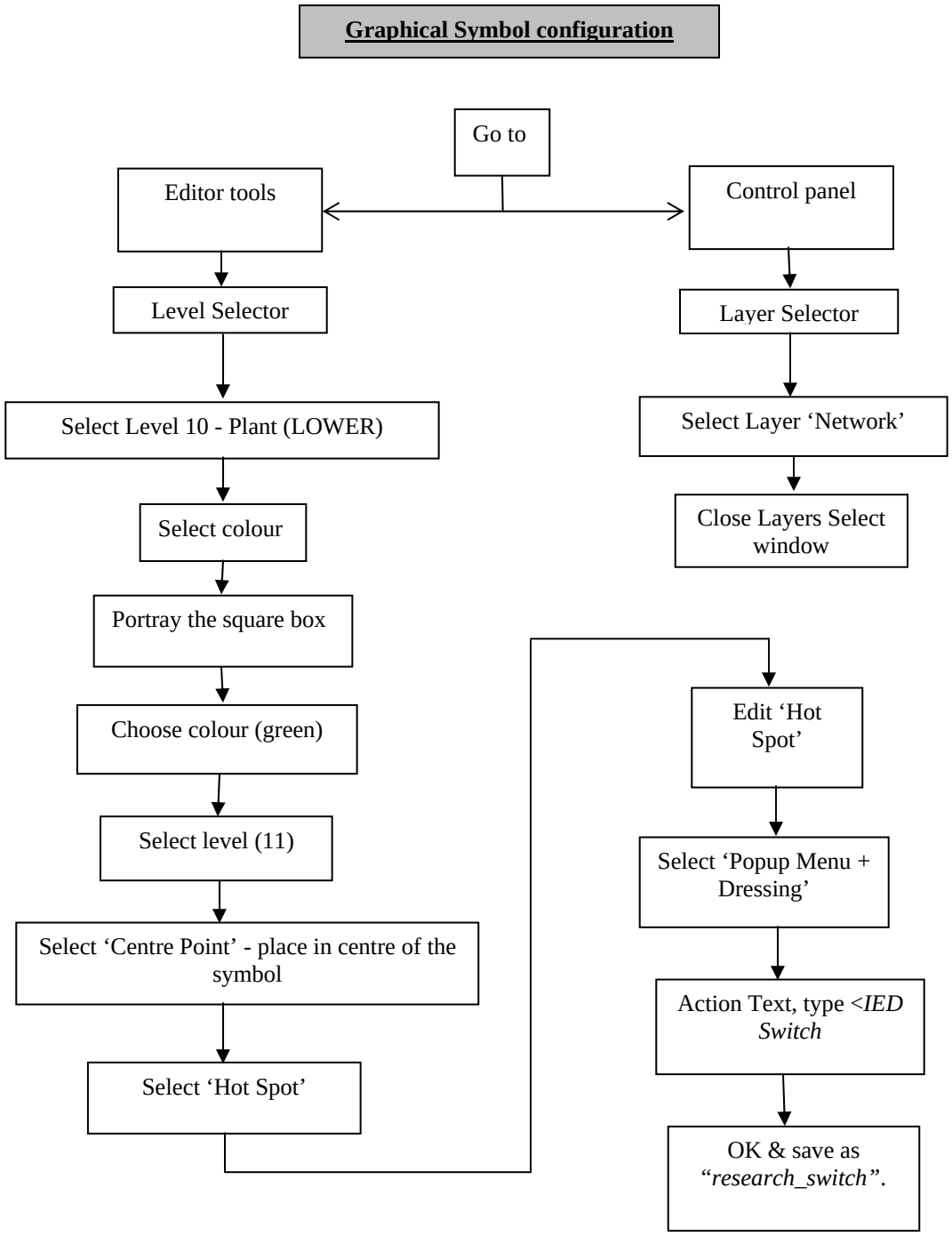


Fig. 5.7. Flow diagram of the graphical symbol configuration process followed during this research.

Fig. 5.8. The Control Panel and Editor Tools respectively.

5.4.4. The Symbol pallet

After the successful completion of the Graphical symbol configuration, the final step was to add this symbol to the symbol pallets for easier retrieval later. By bringing up the Symbol pallet system form via System/Symbols/Symbol Pallets, the following was added:

Table 5.1. Table of the Symbol pallet.

Pallet Name	RESEARCH
Order	Any number that was not used was picked for the purpose of this application.
Symbol Name	research/*research_switch*
Displayed Name	*<IED Switch>*
Level	10
Status	0

The symbol as it stands is a “dumb” symbol at this stage, because it is not tied to a database object yet. What was needed now was to create the database switchgear component.

5.4.5. The Switchgear component

Components not only provide useful data about plant items (e.g. manufacturer, last date inspected, last maintenance date etc.), but they also provide the intelligence behind the graphics. The switchgear component exists as a component in the Plant hierarchy.

5.4.6. Creating a component in RTAP for the research substation

The Component class is used to specify the component's type and also defines the life cycle of the component. All components that were created in this way were initially stored in the 'RDBMS Only' (Relational Database Management System i.e. ORACLE). Figure 5.9 shows the steps that were used during this research to create the component in RTAP.

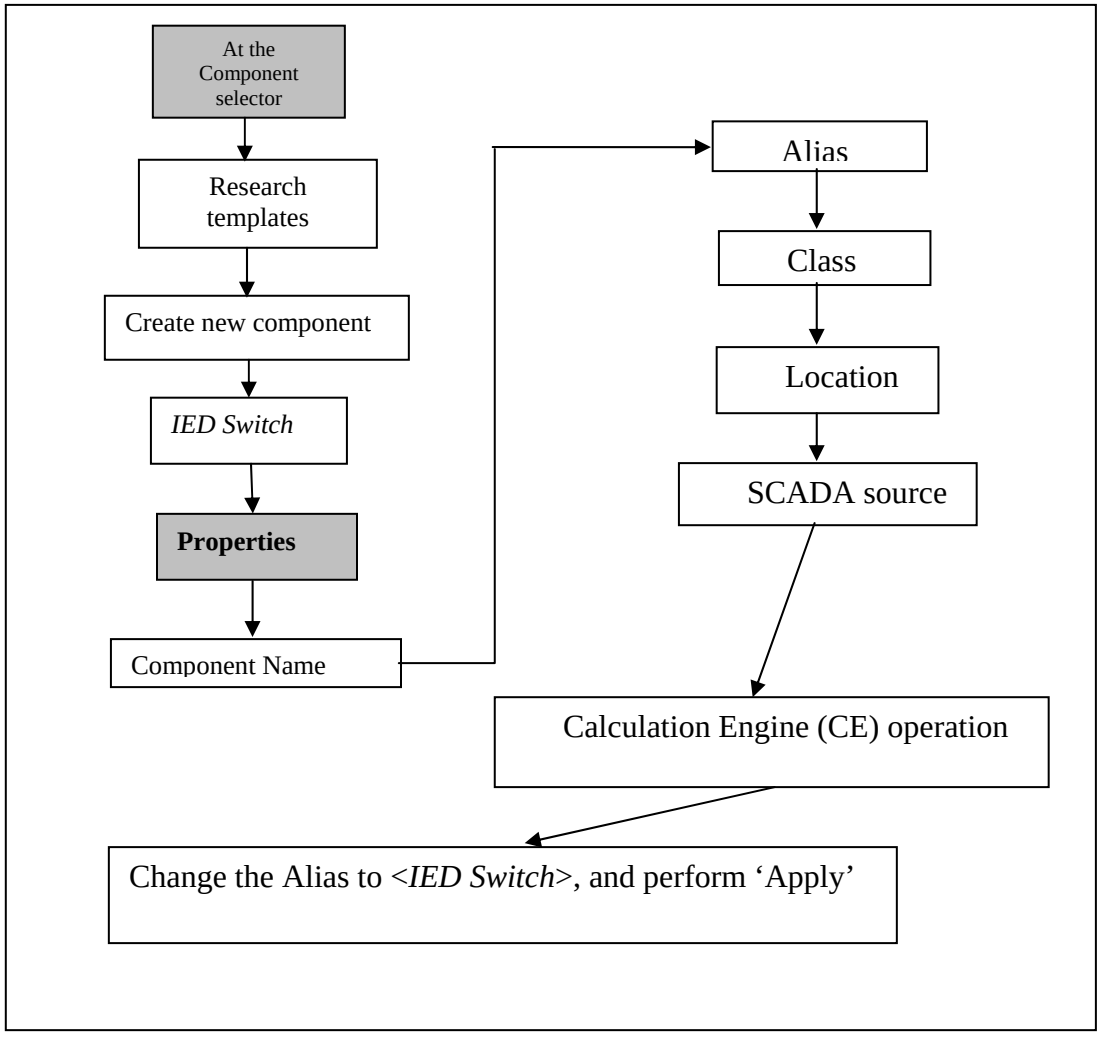


Fig. 5.9. Steps followed during the research to create the component in RTAP.

5.4.7. The Component class

Every component has a class as a property of the component. This class has a variety of uses, e.g.

- For traces to specify components at which to stop (e.g. stop at protection devices).
- For filtering so that all components of a particular class can be identified.

At the **Component class** form, the following details were entered and cached on the client as shown in table 5.2:

Table 5.2. Details of the Component class form.

Index	Class Name	Has Cust	Appearance	Life Cycle	Status
Pick a number between 0 and 999	<***IED Switch***>	N	<***IED Switch***>	<***IED Switch***>	0

5.4.8. The Component Life cycle

The Life cycle of each component was tied to a Component class because every component has a state associated with it and this state is within the Life cycle. In ENMAC, like in many other applications, a number is used to represent a state. This number is actually a bit-field where some of the bits have some significant meanings, as shown in table 5.3.

Table 5.3. Numbers used to represent certain states.

Bit 0	Decimal Value 1	1= open; 0 = close
Bit 1	Decimal Value 2	1 = isolated; 0 = not isolated
Bit 2	Decimal Value 4	1 = earthed; 0 = not earthed
Bit 3	Decimal Value 8	1 = tele; 0 = non-tele

Table. 5.4. Representation of the binary and decimal values.

State	Decimal Value	Binary Value
Manually Closed	0	0000
Manually Opened	1	0001
Tele Closed	8	1000
Tele Opened	9	1001

To move from state to state involves manipulating these bits: for example to move from 'Manually Closed' (0000) to 'Tele Opened' (1001), bits 3 and 0 need to be set. This manipulation of bits was achieved by specifying a *mask* and a *value*. The state transitions are achieved by boolean algebra via the following formula:

new_state = (old_state AND ~mask) OR value

In ENMAC, each transition (i.e. each mask, value pair) is linked to a menu item, and are configured into ENMAC via the System Forms System/Dressing/Life Cycles as shown in table 5.5:

Table 5.5. Life cycle table configured on ENMAC for the purpose of this research.

Name	Current State	Menu	Transition	Next State
<*IED Switch*>	Closed	0	9,1	Opened
<*IED Switch*>	Closed	0	9,9	Teleopened
<*IED Switch*>	Opened	0	9,0	Closed
<*IED Switch*>	Opened	0	9,8	Teleclosed
<*IED Switch*>	Teleclosed	0	9,1	Opened
<*IED Switch*>	Teleclosed	0	9,9	Teleopened
<*IED Switch*>	Teleopened	0	9,0	Closed
<*IED Switch*>	Teleopened	0	9,8	Teleclosed

5.4.9. The Component appearance

The appearance of the component on the diagram is specified by both the Component class as well as the life cycle, and determines what dressing symbols should be applied to what states. This was done by entering the data shown in table 5.6 into the Appearances System form obtained from the System/Dressing/Appearances:

Table 5.6. Table showing the data of the Appearances system form obtained from the System/Dressing/Appearances:

Name	State	Displayed Name	Symbol	Hot Spot Level	Visible Real World	Operational Status	Fore	Back
<*IED Switch*>	0	Closed	research/cb_manual_closed	N	Y	0	0	0
<*IED Switch*>	1	Opened	research/cb_manual_opened	N	Y	0	0	0
<*IED Switch*>	8	Teleclosed	NO_SYMBOL	N	Y	0	0	0
<*IED Switch*>	9	Teleopened	NO_SYMBOL	N	Y	0	0	0

5.4.10. Menus

Menus allow certain actions to be performed on the component the menu is attached to. Early on, the "hot spot" (the area where the pointer gets active when moving over a predefined boundary) was specified to popup a menu called <*IED Switch*>. This menu was then entered into the system form via System/System Configuration/Menus as shown in Table 5.7:

Table 5.7. Table showing the actions performed on the component that the menu is attached to.

Menu Name	Menu Order	Item Name	Function	Data	Status
<*Research*> Switch	10	Examine	EXAMINE_PLANT		0
<*Research*> Switch	20				0
<*Research*> Switch	30	TELE Open	CREATE_OP_FROM_ACTION	<*Research*> TELE CB OPEN	0
<*Research*> Switch	40	TELE Close	CREATE_OP_FROM_ACTION	<*Research*> TELE CB CLOSE	0
<*Research*> Switch	50	Manual Open	CREATE_OP_FROM_ACTION	<*Research*> TELE CB OPEN (MANUAL)	0
<*Research*> Switch	60	Manual Close	CREATE_OP_FROM_ACTION	<*Research*> TELE CB CLOSE (MANUAL)	0

Note that the Data field contains <*Research*> not <*IED Switch*>.

To make a component change state, the CREATE_OP_FROM_ACTION function was used. To drive the component from one state to another, an *action* must be specified. This action defines the mask and value.

5.4.11. Actions

Actions provide the mechanism to perform the state transitions. The *mask* and *value* are specified here. Actions themselves have life cycles and appearances. For each action state, there is a corresponding appearance, as entered by the following into the Actions system form via System/Dressing/Actions as shown in table 5.8:

Table 5.8. Data entered into the Actions system form via System/Dressing/Actions.

Action Name	<*Research*> TELE CB CLOSE (MANUAL)	<*Research*> TELE CB OPEN (MANUAL)
Mask	9	9
Value	0	1
Type	0	0
Appearance	TELE CB CLOSE (MANUAL)	TELE CB OPEN (MANUAL)
Life Cycle	SWITCHING	SWITCHING
Menu Name	Operation	Operation
Displayed Verb	Manually Close	Manually Open
Displayed State	Manually Closed	Manually Opened
Action Name	<*Research*> TELE CB CLOSE	<*Research*> TELE CB OPEN
Mask	9	9
Value	8	9
Type	0	0
Appearance	TELE CB CLOSE	TELE CB OPEN
Life Cycle	TELE SWITCHING	TELE SWITCHING
Menu Name	Operation	Operation
Displayed Verb	TELE Close	TELE Open
Displayed State	TELE Closed	TELE Opened

In the ENMAC system, there are Plant life cycles as well as Action life cycles. Both were entered into the same System form. The action life cycle is also driven from menus, but menus do not need to be explicitly defined for this in the menus configuration form. Instead they are automatically generated based on the life cycle of the action. The 'Menu Name' field specifies a menu to attach to the automatically generated menu and it provides extra menu items that are not derived from the life cycle (e.g. resume job).

5.5. Tying Symbol graphics to the database

The graphics for the switchgear and also the database object behind it were created up to now. The main link between the graphics and the database is through the "hot spot" (the area where the pointer gets active when moved over a predefined boundary).

5.5.1. Creation of the link

The following steps were performed to create the link to the "hot spot".

- Select Create/Link to "hot spot".
- Check that the link has been established by selecting 'Edit Component' from the Editor tools.

This brings up the Attribute summary for the component - this window is similar to the RTAPP display tool. At this stage this window was empty, as no attributes have been added yet.

5.5.2. Attributes

Attributes in ENMAC always exist in ORACLE, but they can also exist in RTAP and/or the Graphic Real Time Values (GRTV). There are four possible combinations:

- **RDBMS only -** for attributes in the ORACLE database only.
- **GRTV only -** for attributes that drive dynamic graphics on the diagram not derived from RTAP e.g. a Power Analysis System.
- **Real Time Database + GRTV -** for attributes that drive dynamic graphics on the diagram from values derived from RTAP.
- **Real Time Database only -** for attributes which have to be stored in RTAP, but do not directly drive graphics

5.5.3. RDBMS only attribute

The attribute summary contains the 'Manufacturer' attribute, and this was edited according to the manufacturer and type of equipment stationed at the substation that was used for the purpose of this research project.

5.5.4. Don't Believe It (DBI) states

This was introduced to avoid the false alarm indication where both error flags are set for the period when no signal is present on the IED. The scan system brings back two bits instead of just one and these are known as double point digitals. Two bits give 4 states, for example:

00 DBI-00
01 OFF
10 ON
11 DBI-11

5.6. Analogues

Analogues can either be returned from the scan system as a "raw" value (e.g. an 8-bit analogue) or in engineering units. We created a generic analogue template that could take any 8-bit analogue and convert it into an engineering unit for measuring the harmonic content of the IED. From this template, the voltage template was modified for the creation of the Harmonic template.

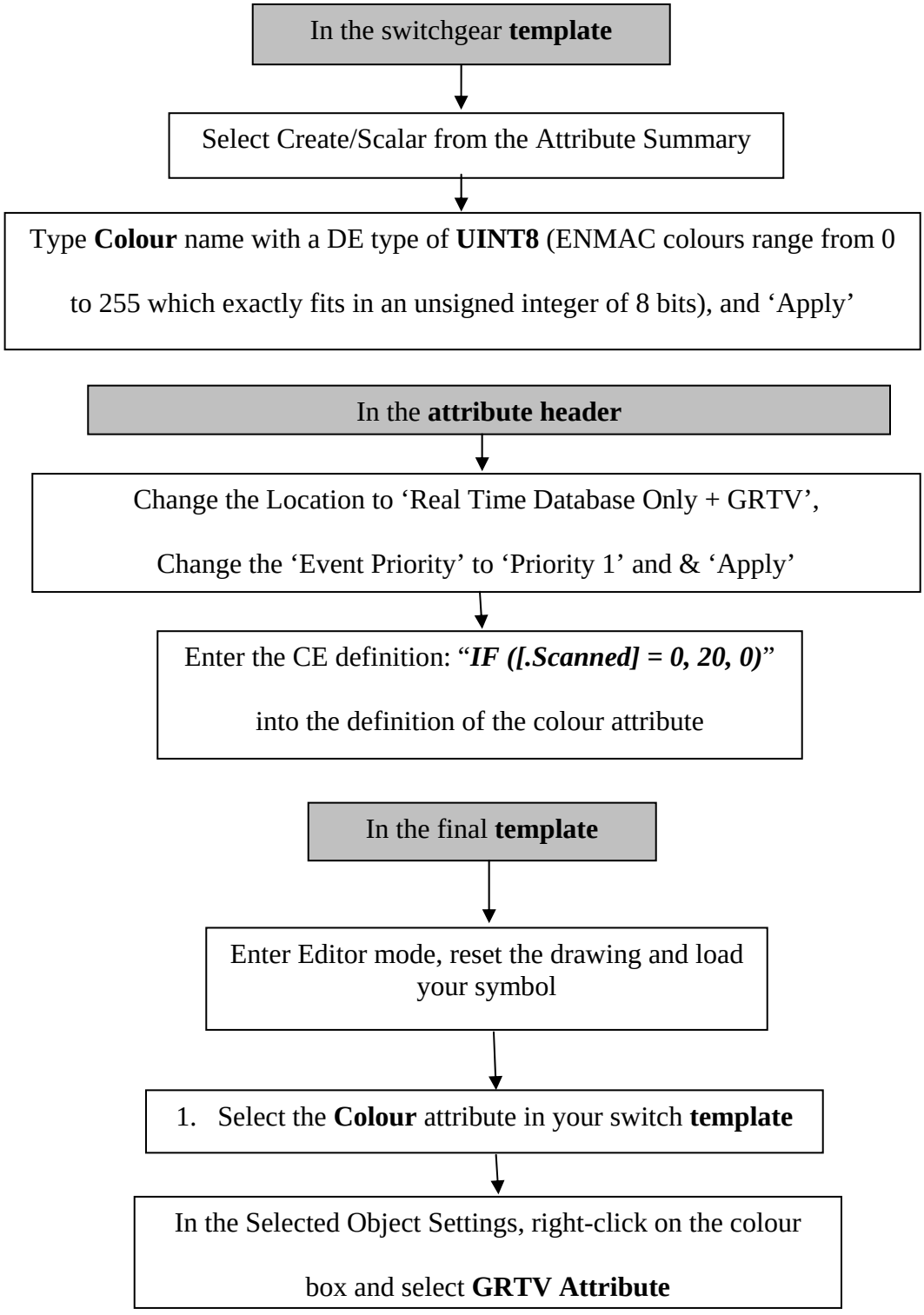


Fig. 5.10. Diagram showing the actions that were performed to create the attribute.

5.6.1. Analogue implementation

For this example of an analogue symbol, the author created the template as shown in figure 5.11.

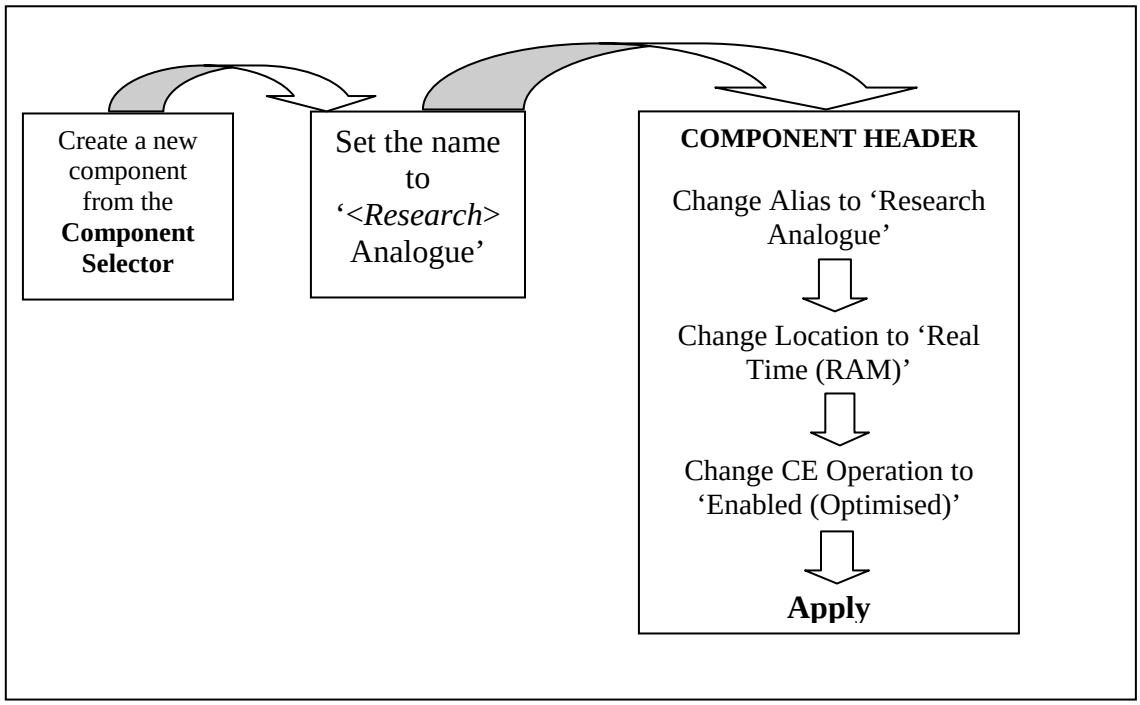


Fig. 5.11. The steps followed during the creation of the analogue template.

5.6.2. The analogue symbol

Up to this point of this research project, the author has created the symbol and attaches it to the analogue component that the author has created for the research substation. The analogue value must be stored in the GRTV as shown in Table 5.9.

Table.5. 9. The analogue value stored in GRTV.

Pallet Name	RESEARCH
Order	Pick a number that is not used.
Symbol Name	research/*research_current*
Displayed Name	<*Research*> Current
Level	10
Status	0

5.6.3. Harmonic reading analogue colour (based on the state)

The following **Scalar** attributes were created in the **TEMPLATE** analogue to allow the harmonic indication colour to be shown correctly:

Table 5.10. Scalar attributes were created to allow the analogue colour to be shown correctly.

Name	DE Type	Location	Priority	Value	CE Definition
High Limit	FLOAT	RTDB	N/A	90	
Low Limit	FLOAT	RTDB	N/A	10	
State Alarm	UINT8	RTDB	N/A		3STATE ([.Display Value], [.High Limit], [.Low Limit])
Default Colour	UINT8	RTDB	N/A	1	
Colour	UINT8	RTDB+GRTV	Priority 1	255	IF ([.State Alarm] <> 1, <red colour>, [.Default Colour])

If we take a brief look at each of these attributes, we discover that:

- **High Limit** and **Low Limit**. This is user specified limits specified by the author that were initially populated with default values and this may be changed for a particular instance.

- **State Alarm**. Uses the **3STATE** function to return the following values: 0 = low, 1 = normal, 2 = high.
- **Default Colour**. The normal colour of the analogue when it is within the range of its limits. It will be set to green for current analogues and yellow for voltage analogues. Note that the default is initially set to 1 (which is white). When the analogue is first placed down, the default will be white and hence we know that it has not been populated with a 'real' default colour yet.
- **Colour**. The colour was tied to the analogue as a dynamic colour, so the location is also in the GRTV.

IF ([.State Alarm] <> 1, 2, [.Default Colour]).

This calculation reads: if the state alarm is not equal to 1 (i.e. it's in low or high), then use the red colour, otherwise use the default colour.

5.7. IED building procedure on the ENMAC master station

After completion of the Actions, Appearances, Lifecycles, Analogues and other essentials as explained during this chapter, the actual construction of the IED device outputs followed. Firstly the author had identified where to place the actual station diagram on the ENMAC visual world. To know where to start on the display, the X and Y co-ordinates were found because in ENMAC, different worlds exist, e.g. for the Reticulation

stations, we us the “Retic” world, for the Distribution stations, we use the “Distr” world, and for Transmission stations the “Tx” world.

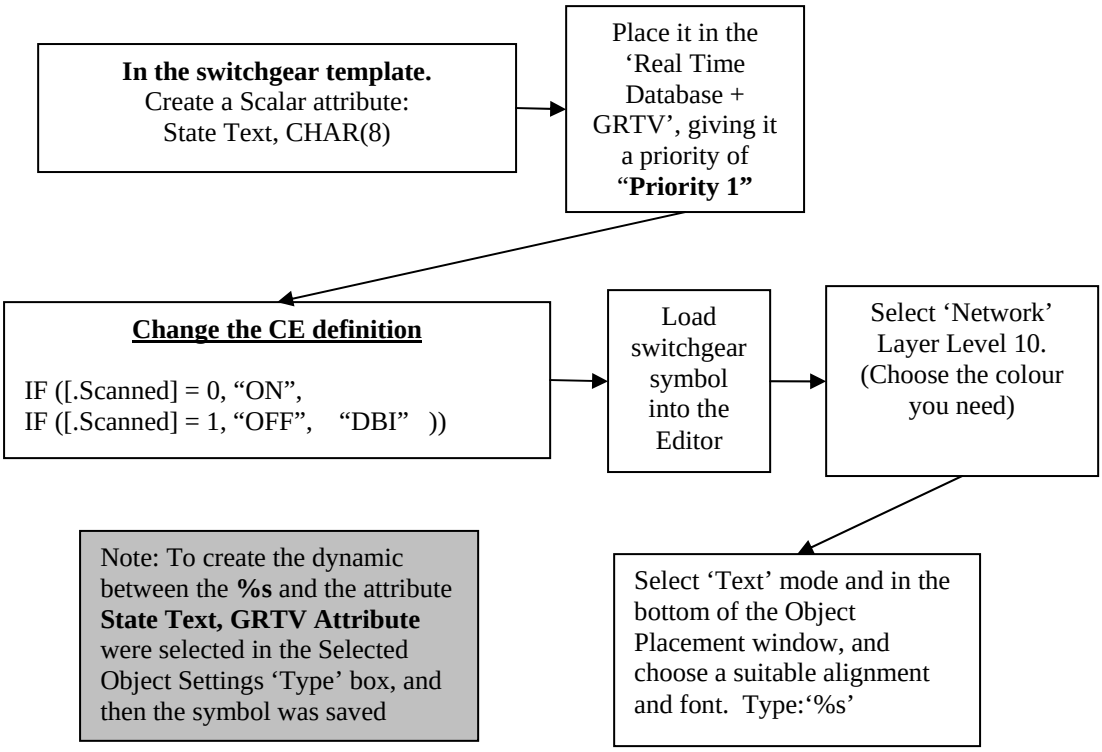


Fig. 5.12. Example of the string placed in the calculation engine (CE) to declare the alarm state ON or OFF.

5.7.1. System configuration

When the desired location of the substation area was found, the name of the substation was placed in an Oracle lookup table. The lookup table is used to define sets of values that are accessed and referred to in various ways within ENMAC.

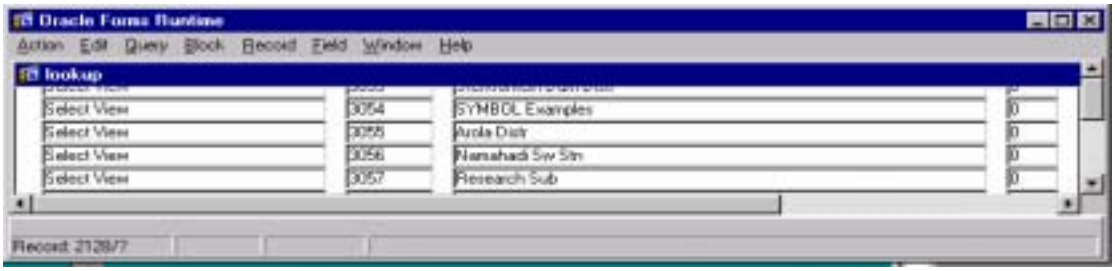


Fig. 5. 13. The Oracle lookup table.

After placing the substation name in the correct hierarchy in the database, the Load Symbol option from the Editor menu allows the previously created symbols to be loaded into the Editor. Then the Symbol pallet window displays the available symbols for the currently selected symbol pallet.

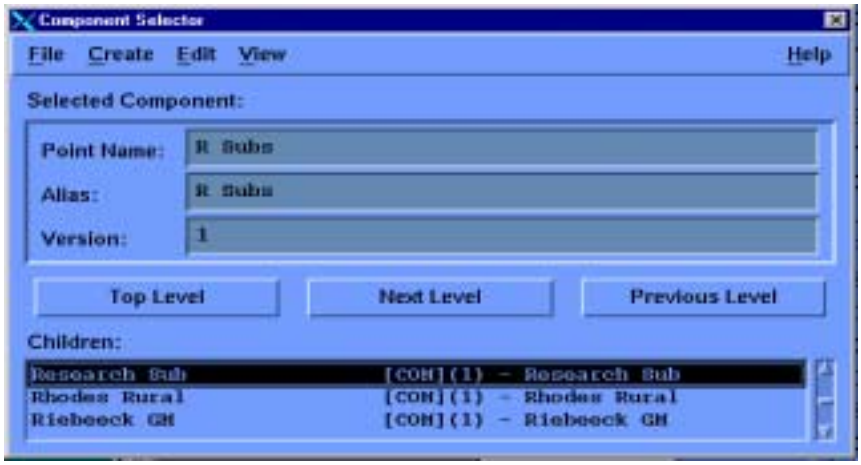


Fig. 5. 14. The component selector showing the research substation location.

5.8. IED hierarchy

In the picture shown in figure 5.15, the reader can see that the harmonic components as well as the dip and surge measurements were categorised under the general alarms structure for the IED. The rest of the plant items can also be viewed.

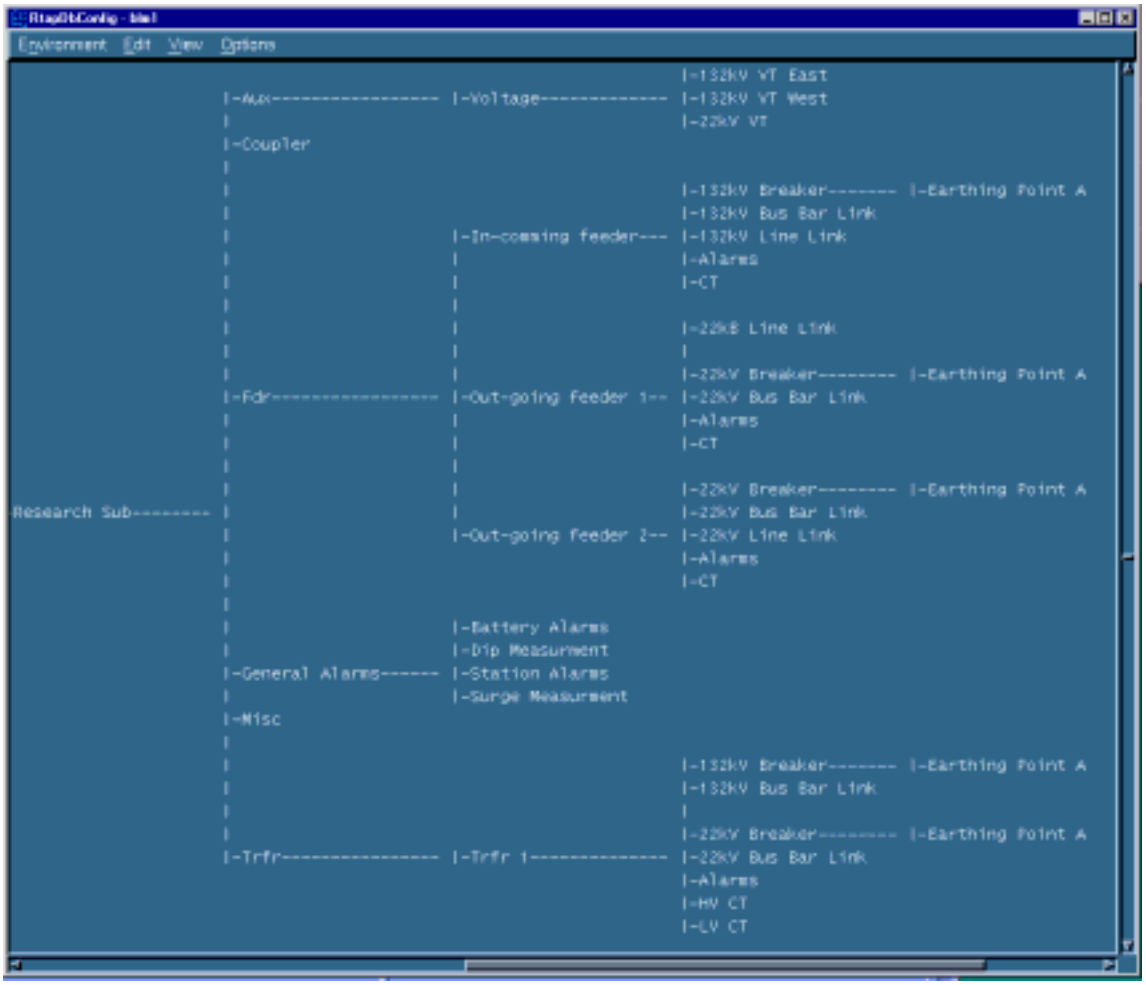


Fig. 5.15. RTAP view of the substation hierarchy, showing the RTAP configuration.

Figure 5.16 shows the Plant item table as compiled and used for the research project.

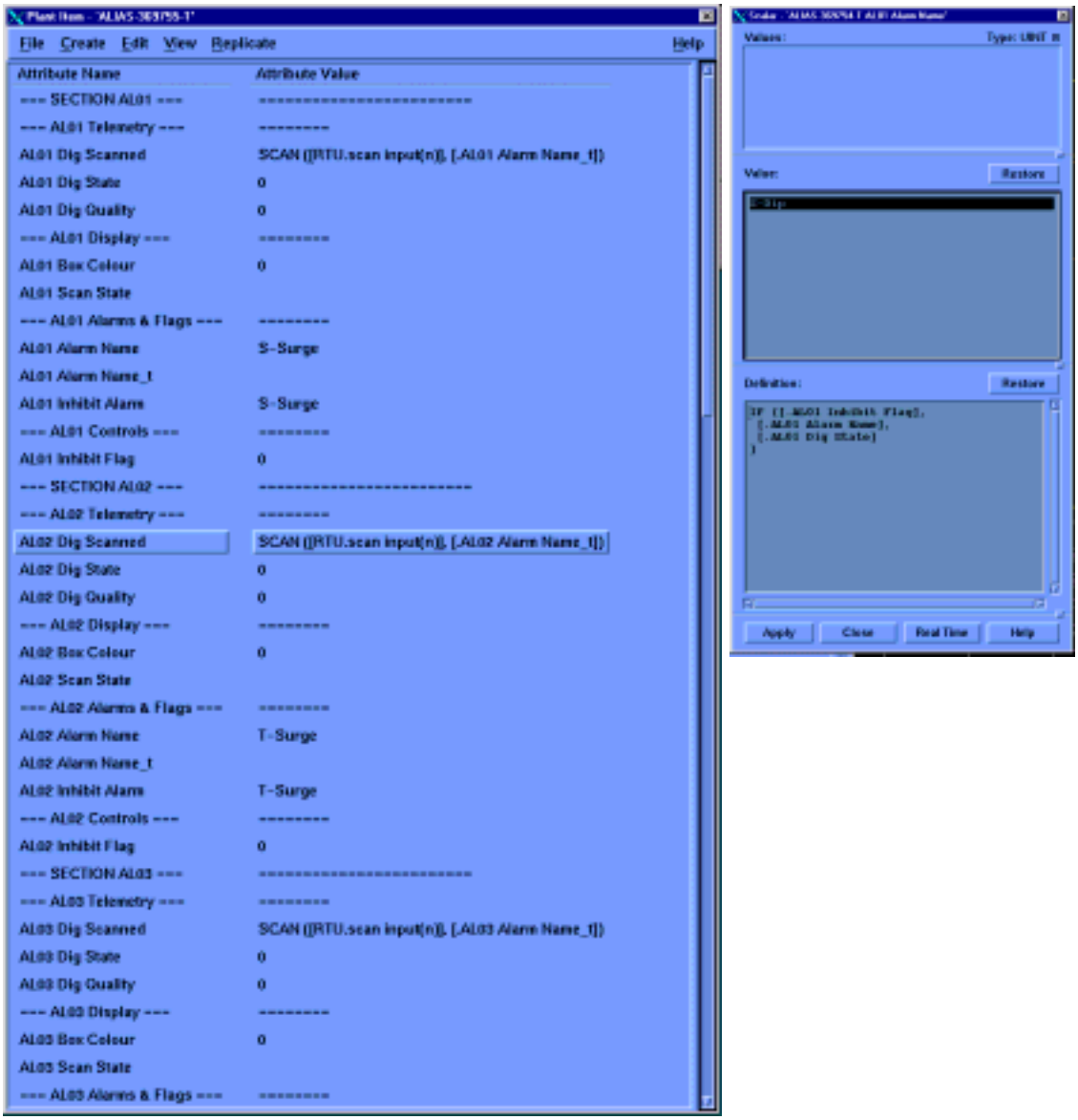


Fig. 5.16. The plant items table as compiled and used for the research project.

The following information shows the values used by the author for harmonic 1 (Fundamental):

- For the symbol to change state according to the symbol Box Colour:

  IF ([.AL02 Inhibit Flag]=1,6,

  IF ([.AL02 Dig State]=1,2,3))

- Harmonic 1 (Fundamental) Scanned value:

  ABS([.Harm1 Scanned Val] * [.Calibration Factor] / ([.Scale Value] * 1000))*100

  - where ABS = Absolute value.

  Harmonic 1 (Fundamental) QUALITY ([.Harm1 Scanned Val])

- Harmonic 1 (Fundamental) High Limit, where the high limit must be set when a value >100% was reached; the author used the NRS value as a reference: [.NRS Value]*1.001
- The same applies for the Low limit: Harmonic 1 (Fundamental) Low Limit [.NRS Value] * 0.999. In other words, any value <>1 will be a limit state, either high or low.
- Harmonic 1 (Fundamental) Alarm Limit State: IF ([.Harm1 Disable Tele] =0, [.Harm1 Limit State], 3STATE ([.Harm1 Formulae], [.Harm1 High Limit], [.Harm1 Low Limit])

Note: - Harm1 Limit State_t is used for time tagging, where the actual RTU event time is used and send to the Master for time stamping the event. When we look at the state settings, showing that for state 0 and 2, where state 0 is a non-alarm state (low colour), Low level, and state 2 is a alarm state (high colour), High level.

Harmonic 1 (Fundamental) Display Colour:

IF ([.Harm1 Disable Tele]=0 OR [.Harm1 Quality]=3,6,

IF ([.Harm1 Quality]<>0,9,

IF ([.Harm1 Limit State]=0,[.Low Colour],

IF ([.Harm1 Limit State]=2,[.High Colour],[.Normal Colour]))))

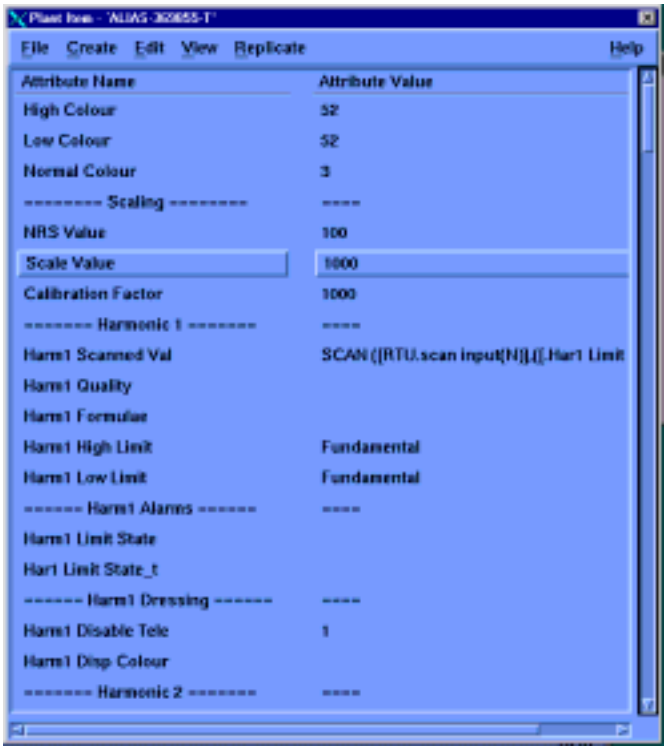


Fig. 5.17. The fundamental (Harm 1) settings as configured on ENMAC.

♦ Exactly the same criteria were used for harmonics 2-13 and the Total Harmonic Distortion, RMS and Peak harmonics on UNIX.

5.8.1. IED alarm definitions

To manage a network efficiently and safely, an operator must be aware of abnormal conditions as soon as possible. The ENMAC system provides an alarm facility that draws any predefined condition to an operator's attention automatically.

An alarm is associated with one or more component attributes. When the attribute value equals that which the alarm is set to detect, a message appears in the appropriate alarm display window. The alarm definition table is used to configure the following:

- Alarm value
- Alarm text name
- Alarm priority
- Alarm type
- Acknowledge class
- Acknowledge action

Figure 5.18 shows a picture of the Alarm definition tables as configured for the IED.

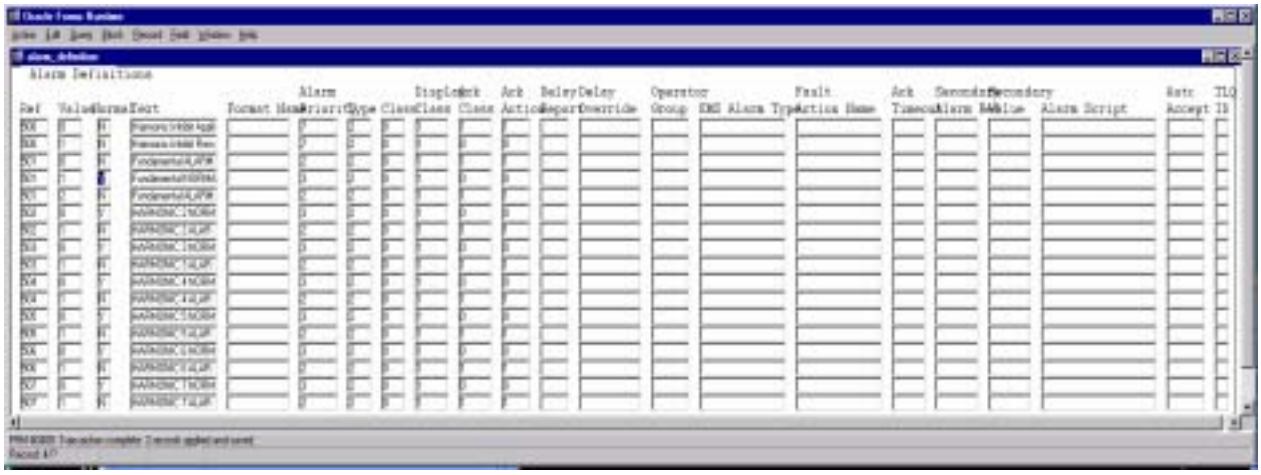


Fig. 5.18. The alarm definition table.

To verify that all settings were replicated into the RTAP, a display of the RTAP table was made before testing. Figure 5.19 below shows the RTAP display with all the values that were explained so far to be observed by the reader.

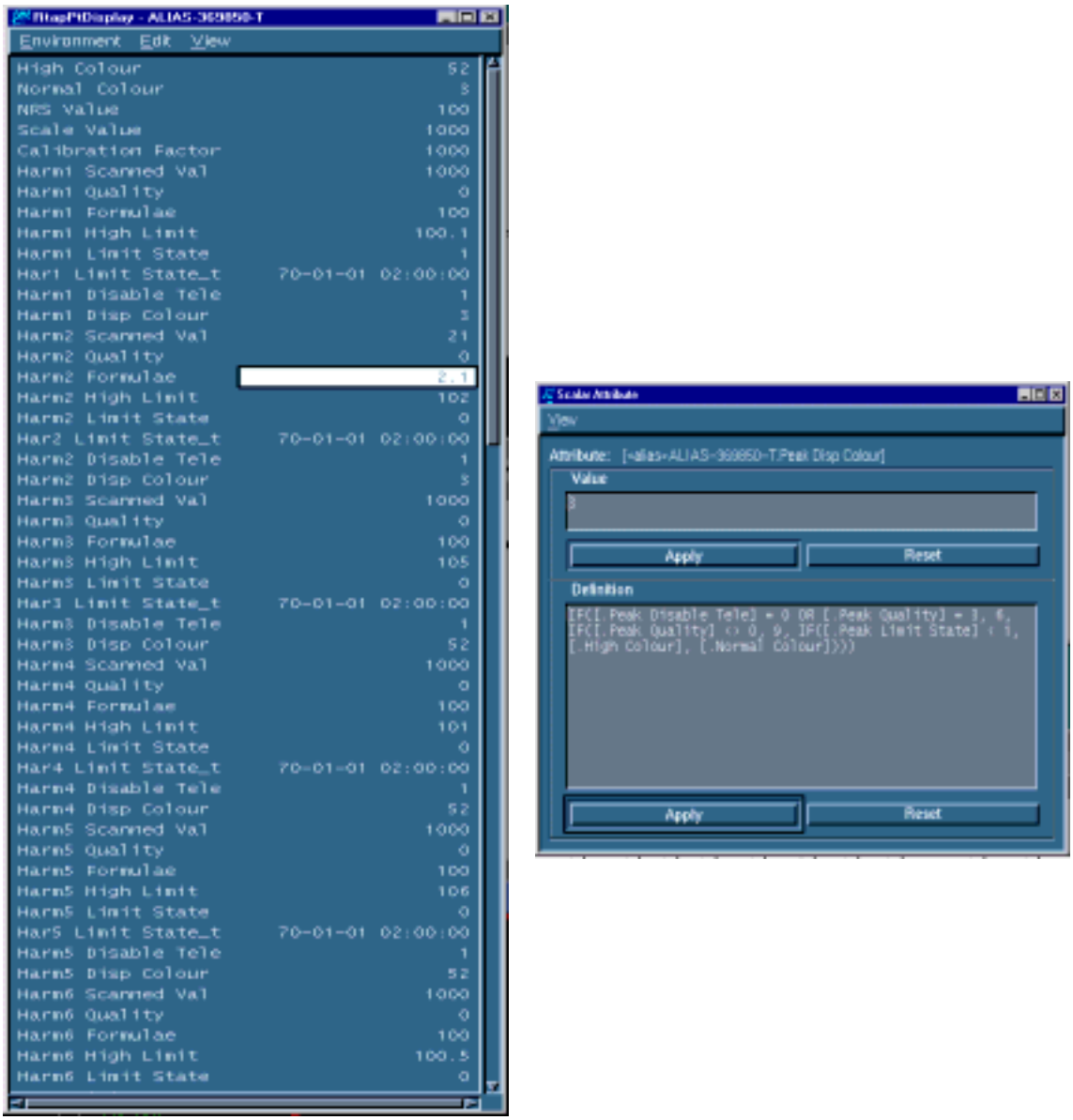


Fig. 5.19. The RTAP display showing some of the values used for this research.

Following the successful completion of the database configuration**,** all the scan input and output points were linked to the symbols on the research substation drawing. All alarms,

like dip and harmonic categories, are shown in the alarms page [Fig. 6.34, p. 220]. The next step was to establish communications with the RTU at the substation. A picture of the research project substation layout as viewed by the ENMAC operator is shown in figure 5.20.

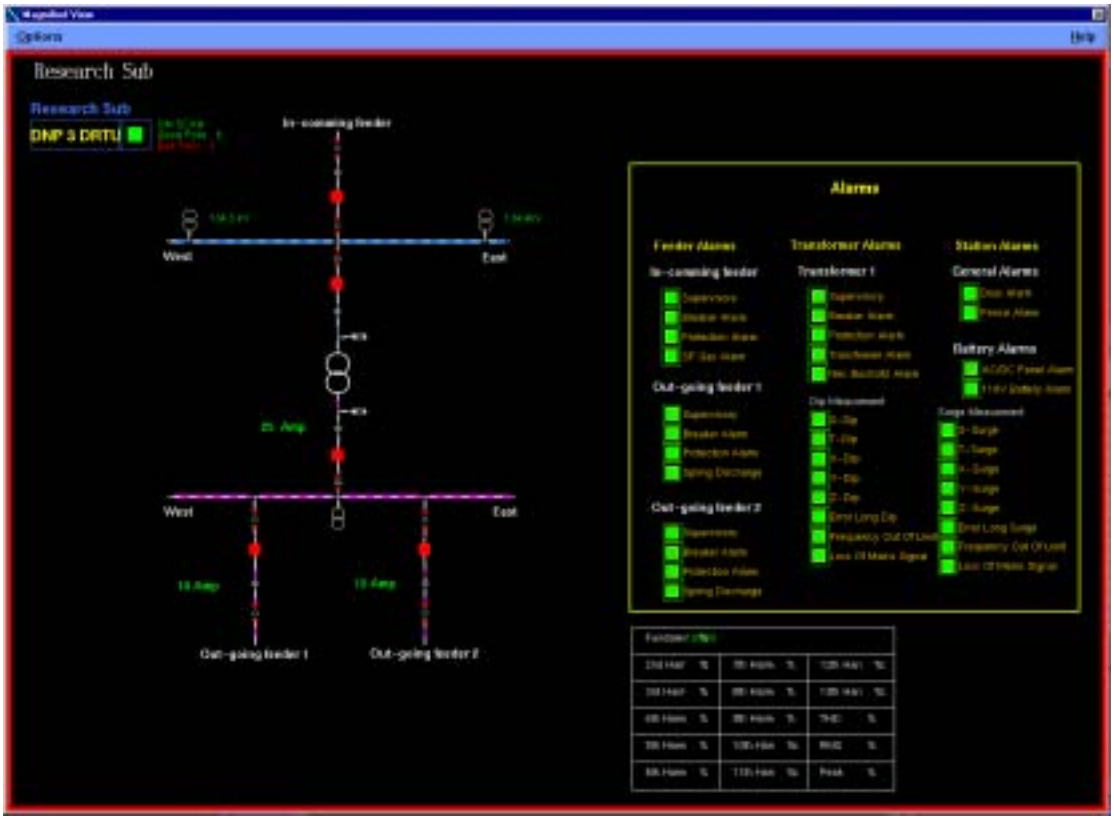


Fig. 5.20. The complete substation layout, as developed for this research project for the ENMAC system.

All relevant commissioning, implementation and reports are discussed in chapter 6.

5.9. Summary

To process the measurements of the Intelligent Electronic Device (IED) at the substation, the existing SCADA system, called the ENMAC was incorporated. This consists of the Client, Network Management Server (NMS), SCADA server and the Front End Processor (FEP). The main types of interactions between the NMS and the SCADA server used are the transactions, events, telecontrol, and SCADA commands.

The data that were generated by the IED development were stored at the three main data stores on the ENMAC:

1) The Real Time Database (RTDB) which exists on the SCADA server.
2) The ORACLE database exists on the NMS server.
3) The Graphic Real-Time Values are kept at the NMS server.

The Oracle database programme was extensively used during this development to maintain the whole plant database hierarchy, while RTAP only contains a subset of the plant database hierarchy. The Scan Task (on the SCADA server), Distfep, (on the SCADA server), and Netfep (on the Front End Processor) are responsible for intelligent polling of all the RTU's.

A suitable explanation on the ENMAC configuration, like the Oracle database configuration, RTAP (Real Time Application) database and GRTV (Graphic Real Time

Values) were discussed in this chapter. To understand the state transitions, the reader was given an overview of the action configuration processes, showing that actions themselves have life cycles and appearances, initially created in a schedule.

ENMAC monitors all component attributes that can activate an alarm. Whenever an attribute value changes the new value is compared with the value fields of the alarm definitions matching alarm reference numbers. Remarkably, a defect was detected on the ENMAC system upon entering data during this research. The author found that sometimes the RTAP tables were not updated automatically. This imposed the author to individually examine each input on RTAP, to ensure the RTAP application have been accepted and updated to guarantee successful configuration actions.

Even though during this chapter the author intended to cover a large amount of the most relevant development configurations, it was found impracticable to give detailed information to the reader due to the complexity and combination of the systems in general. The author has, therefore, attempted to enlighten the most important and basic features used, to enable the reader to be able to comprehend the technique of this phase of the development.

The successful configuration that was completed during this chapter was then used to represent the final research substation, to facilitate the official IED development tests as discussed and shown in chapter 6.

CHAPTER 6

HARDWARE & SOFTWARE EVALUATION

6.1. Introduction

This section describes the evaluation and the results that were obtained on all software and hardware devices during this research. This includes the RTU; the IED bread boarded version, the final IED development as well as the ENMAC interfacing configurations. During this section, most of the evaluation tests were performed locally by using the following measuring and monitoring equipment:

1) The FieldComm Program.
2) Hewlett Packard Spectrum Analyzer 3582A analogue analyzer.
3) VectoGraph measuring system.
4) Video Terminal DOS program.
5) Omicron CMC 156 test set.
6) ENMAC SCADA Master Station.

6.2. Operational testing of the system as developed on the bread boarded prototype

During the operational testing stage of the bread boarded prototype, a Pentium-class PC was introduced at this stage so that we could test the programs. The PC was also used to mathematically generate a set of samples. These generated samples, working on LINUX, were then injected into the program itself, to test proper operation on the prototype bread boarded version.

6.2.1. Wave detection tests

The C program *anal.c* [App.1, p. 248] was designed to analyse the given wave form on a PC. The program reads 16-bit hex values in the format that is output by *wave.ccp*. It implements a function called "*detect*", which is a Discrete Fourier Transform for the harmonics that we wished to detect.

Our wave samples were periodic, and since our sampling rate was low, we implemented a Discrete Transform, which is a lot faster than the more usual FFT under these conditions (FFT is also discrete, but it's not really exact to state that it's faster, as it uses a sliding window). Speed was an issue for us, as a more efficient algorithm allows a smaller processor to be used in production.

All development was done in LINUX, to allow the output of *wave.ccp* to be connected to the input of *anal.c* by use of a simple pipe. *Anal.c* was able to detect the harmonics of *wave.ccp* both correctly and accurately.

The results are shown in table 6.1:

Table 6.1. Table of the detected and generated sine wave with a frequency of 50 Hz.

Harmonic	Freq % Value Generated	Value Detected	Generated %
0	**11.9401**	**100.00%**	**100.00%**
1	0.00023	0.00%	0.00%
2	0.00031	0.00%	0.00%
3	0.0005	0.00%	0.00%
4	**0.95587**	**8.01%**	**8.00%**
5	0.00097	0.01%	0.00%
6	0.00141	0.01%	0.00%
7	0.00287	0.02%	0.00%
8	**2.62738**	**22.01%**	**22.00%**
9	-0.0015	-0.01%	0.00%
10	-1E-05	0.00%	0.00%
11	0.00148	0.01%	0.00%
12	**0.47735**	**4.00%**	**4.00%**
13	-0.0018	-0.02%	0.00%
14	-0.0005	-0.01%	0.00%
15	0	0.00%	0.00%

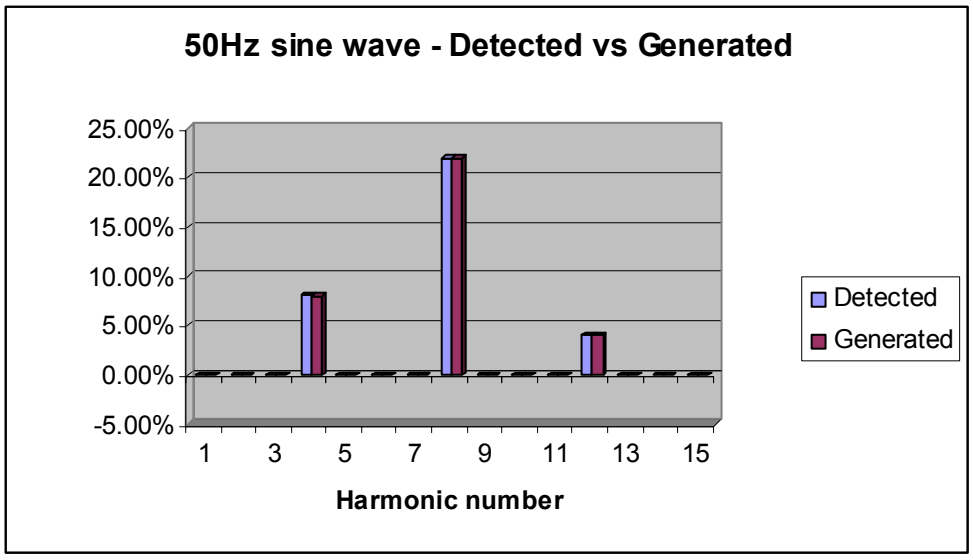


Fig. 6.1. Graph of the 50Hz detected and generated sine wave.

The reader will observe that the detection is excellent – with errors typically better than 0.01%, which was more than good enough for our requirements. If we required higher resolution, we would need a sample size of bigger than 16 bits, but these figures indicate that this was unnecessary. Table 6.1 was produced with the two programs *wave.ccp* and *anal.c* exactly in phase. Subsequently *Anal.c* was modified to ignore the first few samples in order to check phase behaviour. Table 6.2 and 6.3 shows some of the outputs:

Table 6.2. Table of the generated and detected fundamental for 50 Hz, phase shift between *wave.ccp* and *anal.c* is 1/8 period.

Harmonic number	Freq. (Hz) % Generated value	Detected %
harmonic 0.	Value 8.442895	100.000%
harmonic 1.	**Value 0.000074**	**0.001%**
harmonic 2.	**Value 0.000135**	**0.002%**
harmonic 3.	**Value 0.000183**	**0.002%**
harmonic 4.	Value -0.675190	-7.997%
harmonic 5.	**Value 0.000338**	**0.004%**
harmonic 6.	**Value 0.000484**	**0.006%**
harmonic 7.	**Value 0.001434**	**0.017%**
harmonic 8.	Value 1.850500	21.918%
harmonic 9.	**Value -0.002130**	**-0.025%**
harmonic 10.	**Value -0.001518**	**-0.018%**
harmonic 11.	**Value -0.001953**	**-0.023%**
harmonic 12.	Value -0.332919	-3.943%
harmonic 13.	**Value 0.000902**	**0.011%**
harmonic 14.	**Value 0.000257**	**0.003%**
harmonic 15.	**Value 0.000000**	**0.000%**

Table 6.3. Table of the generated and detected fundamental for 50 Hz, phase shift between *wave.ccp* and *anal.c* is 1/2 period.

Harmonic number	Freq. (Hz) % Generated Value	Detected %
harmonic 0.	Value -11.940093	100.000%
harmonic 1.	**Value -0.000248**	**0.002%**
harmonic 2.	**Value -0.000255**	**0.002%**
harmonic 3.	**Value -0.000565**	**0.005%**
harmonic 4.	Value -0.955882	8.006%

harmonic 5.	**Value -0.001091**	**0.009%**
harmonic 6.	**Value -0.001433**	**0.012%**
harmonic 7.	**Value -0.002808**	**0.024%**
harmonic 8.	Value -2.627399	22.005%
harmonic 9.	**Value 0.001606**	**-0.013%**
harmonic 10.	**Value 0.000001**	**-0.000%**
harmonic 11.	**Value -0.001420**	**0.012%**
harmonic 12.	Value -0.477295	3.997%
harmonic 13.	**Value 0.001716**	**-0.014%**
harmonic 14.	**Value 0.000531**	**-0.004%**
harmonic 15.	**Value -0.000000**	**0.000%**

Table 6.3 shows that even if the simulated signal was exactly out of phase with the evaluation unit, the resultant output was consistent.

Although the tables above illustrate the points that are relevant for our present purposes, the author gives one more table showing how the algorithm performs at 50Hz and with a bigger content of low-order harmonics, which represents a typical real-world case, as in table 6.4.

Here is the harmonics set:

```
main()
{
  /* set the amplitude of the harmonics that we want */
  /* set one frequency to 100 (fundamental, usually) and
  then the others to a percentage of that.  A scaling
  factor to avoid overflow will be calculated later */

  harmonic_amplitude[0]=100;
  harmonic_amplitude[1]=20;
  harmonic_amplitude[2]=15;
  harmonic_amplitude[3]=12;
  harmonic_amplitude[4]=10;
  harmonic_amplitude[5]=8;
  harmonic_amplitude[6]=7;
  harmonic_amplitude[7]=6;
  harmonic_amplitude[8]=5;
```

Table 6.4. Table of the generated and detected sine wave for 50 Hz.

Harmonic number	Freq. (Hz) % Generated value	Detected %
harmonic 0.	Value 8.743033	100.000%
harmonic 1.	Value 1.749048	20.005%
harmonic 2.	Value 1.311490	15.000%
harmonic 3.	Value 1.049025	11.998%
harmonic 4.	Value 0.874627	10.004%
harmonic 5.	Value 0.699674	8.003%
harmonic 6.	Value 0.611639	6.996%
harmonic 7.	Value 0.524461	5.999%
harmonic 8.	Value 0.436834	4.996%
harmonic 9.	Value -0.000595	-0.007%
harmonic 10.	Value -0.000387	-0.004%
harmonic 11.	Value -0.000201	-0.002%
harmonic 12.	Value -0.000145	-0.002%
harmonic 13.	Value -0.000082	-0.001%
harmonic 14.	Value 0.000012	0.000%
harmonic 15.	Value 0.000000	0.000%

These results were consistent with our expectations; very good, as long as the source fundamental really was 50 Hz.

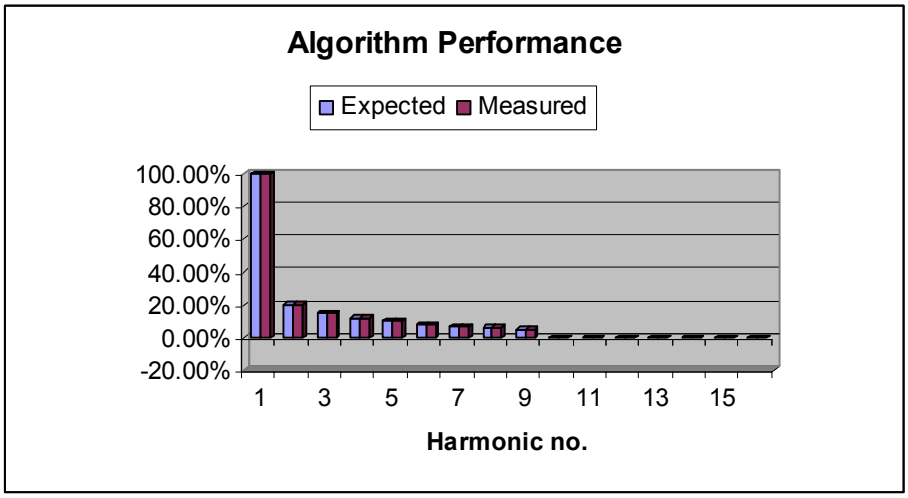


Fig. 6.2. Algorithm performance with source fundamental of 50Hz.

However, in real-life, the mains frequency does not remain at exactly 50 Hz. *Wave.ccp* was recompiled setting the fundamental frequency to 47 Hz, keeping the sampling rate at 625

microseconds, instead of adjusting to 665 microseconds. Here are the new results (Table 6.5):

Table 6.5. Table of the generated and detected sine wave for 47 Hz.

Harmonic number	Freq. (Hz) % Generated value	Detected %
harmonic 0.	Value 12.005977	100.000%
harmonic 1.	Value 0.865608	7.210%
harmonic 2.	Value 0.451508	3.761%
harmonic 3.	Value 0.183149	1.525%
harmonic 4.	Value 0.795511	6.626%
harmonic 5.	Value 0.381909	3.181%
harmonic 6.	Value 0.324394	2.702%
harmonic 7.	Value 0.431562	3.595%
harmonic 8.	Value -0.014051	-0.117%
harmonic 9.	Value 0.093680	0.780%
harmonic 10.	Value 0.125652	1.047%
harmonic 11.	Value 0.390919	3.256%
harmonic 12.	Value -0.049439	-0.412%
harmonic 13.	Value -0.004914	-0.041%
harmonic 14.	Value 0.000877	0.007%
harmonic 15.	Value 0.000000	0.000%

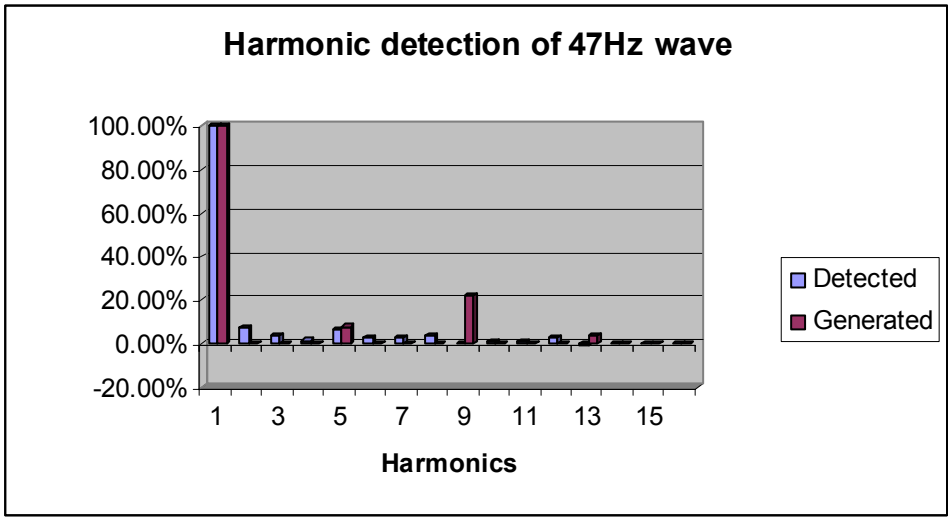


Fig. 6.3. Harmonic detection on a generated 47Hz wave.

The results were very different – we were detecting harmonics that were not there and were ignoring harmonics that existed. The problem worsens for the higher harmonics, which was

understandable, as the time offset was a bigger proportion of the wave for these higher frequencies.

From these results, the author concluded that the algorithm needs to somehow synchronise itself to the fundamental frequency in order for its results to be reliable.

In practice, Eskom's frequency drift is very small; a drift of 0.3Hz from nominal will cause an alarm at both Eskom Generation and Transmission Control centre, and immediate action will be taken. This 0.6% deviation is small enough to ignore in the case of the fundamental, but a 0.6% deviation to the fundamental translates to 13 times that amount – nearly 8% - for the 13th harmonic. Although Eskom will regulate their frequency for an overall average of 50Hz, the adjustments are too slow for our purposes; we have to synchronise to a periodic wave as we conclude that a fixed period between samples is unacceptable, and a solution were necessary.

6.2.2. Solutions to frequency drift

There were two potential solutions to the problem of frequency drift, e.g. to adjust the sampling rate or to compensate to a new fundamental:

- **Adjust Sampling rate**. Using a fixed sampling rate of 625 microseconds only works with 50Hz, as we get a cycle repeat after exactly 32 samples. If we were to adjust the sampling rate to always equal 1/32 of the fundamental period, the harmonics would always appear in the correct positions relative to the sample. For

instance, the correct sampling rate for 47 Hz would be about 664 microseconds. We repeated the 47 Hz test, re-sampling at 664 microseconds.

The results are shown in table 6.6:

Table 6.6. Generated sine wave is 47 Hz, re-sampled to 664 microseconds.

Harmonic number	Freq. (Hz) % Generated Value	Detected %
harmonic 0.	Value 11.945952	100.0%
harmonic 1.	Value 0.017209	0.144%
harmonic 2.	Value 0.005344	0.045%
harmonic 3.	Value -0.003012	-0.025%
harmonic 4.	Value 0.953892	7.985%
harmonic 5.	Value 0.000442	0.004%
harmonic 6.	Value -0.009425	-0.079%
harmonic 7.	Value -0.027772	-0.232%
harmonic 8.	Value 2.626130	21.983%
harmonic 9.	Value 0.030606	0.256%
harmonic 10.	Value 0.013069	0.109%
harmonic 11.	Value 0.003702	0.031%
harmonic 12.	Value 0.483031	4.043%
harmonic 13.	Value 0.008798	0.074%
harmonic 14.	Value 0.003435	0.029%
harmonic 15.	Value 0.000000	0.000%

The results were consistent with the author's expectations and were close to those of table 6.1. The slight discrepancies were as a result of our sampling rate, 664 microseconds, which was not an exact factor of the period.

- Compensate to new Fundamental. If the sampling rate remained fixed, it would be necessary to alter "*anal.c*" in order to detect the fundamental and adjust its calculations accordingly. This route is only stated, but was not

explored at that time, because the first solution, the adjustment of the sampling rate, seemed better and easier to implement in hardware for our purpose.

6.3. Recording Total Harmonic Distortion (THD)

Once the author has determined and stored the harmonics – the calculation of Total Harmonic Distortion was trivial. We know that the distortion factor, as defined in chapter 2 [Ch2, p. 30] is described as:

$$THD = \sqrt{\frac{Sum\_of\_squares\_of\_amplitudes\_of\_all\_harmonics}{Square\_of\_amplitude\_of\_fundamental}} x100\% \quad \text{----------(6.1)}$$

Therefore, using the wave simulator with a 100% fundamental,

8% of 5^{th} harmonic,

22% of 9^{th} harmonic and

4% of 13^{th} harmonic,

Then the THD calculates to 23.1%.

The author then ran a pure 50Hz sine wave through the code, and it detected a 0.0012% THD. This was a constant value, not random. This was due to quantisation error, and is almost always present in sampling applications [38, p. 37]. Its magnitude depends on the sampling accuracy and the calculation accuracy. In this case, we were using 16-bit word sizes for the samples. A one-bit rounding error on a signed 16-bit quantity represents an error of 0.00004% for each sample. If each rounding error was in the same direction, this would give a worst case error of 0.012%. (300 samples X 0.00004% error).

The actual case would be much better than this (as illustrated by the simulation), since it is highly unlikely that 300 successive errors will be in the same direction - some will be too high, some too low and they would tend to average out. But even the worst case is acceptable for our purposes. Increasing the word length from the industry standard of 16 bits to improve this figure increases complexity with very little additional benefit. The filter could adjust the first 10 000 samples, after which there was no more filtering.

6.4. Dip simulation

To simulate dips, a LINUX program called "*filter*" was written, whose sole purpose was to modify the amplitude of the samples that pass through it. When placed after the wave program, it was used to reshape the output of wave with the required dip information. *Filter* reads dip information from a file specified as a command-line argument. The file contains the filtering profile information in a simple language that is defined as follows:

- Lines starting with # are ignored – they are used for comments.
- Lines starting with a non-digit are assumed to be garbage and are also ignored.
- 75 27 means adjust amplitude to 75% on the next 27 samples.
- 55 means adjust amplitude to 55% on the next sample.

The filter can adjust the first 10'000 samples, after which there is no more filtering. Here is an example file:

After one complete cycle, drop-out for one and a half more cycles

25 5
87 10
100 9
55 6

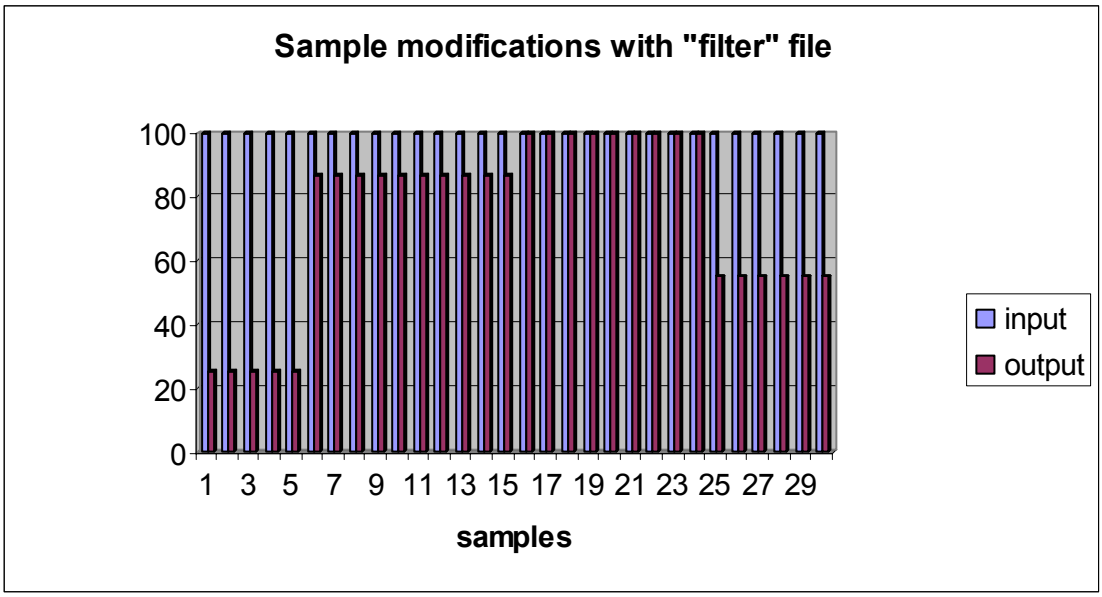


Fig. 6.4. Graph showing how the filter file modifies a set of samples.

Figure 6.4 shows the input samples (all been set to 100%), and the output, after being modified by the filter parameters above - the first 5 samples are cut to 25% of their input amplitude, the next 10 samples are at 87%, then there are 9 samples which are at 100% of the input (i.e. unmodified) and the last 6 samples are at 55%. It is clear that by using this simple configuration file to adjust amplitudes, dips and surges of all magnitudes and durations can be simulated, by modifying the amplitudes of the samples.

6.4.1. The dip detection routine

A file (shell script) called "*prog*" was used to test dips and surges. Table 6.7 shows some of the cases that were tested, along with the measured outcome. It should be noted that this table shows some of the more interesting tests only – those that occur near the boundary conditions. The values that fall well into a dip region were correctly detected and are not showed in the table.

Table 6.7. Table of cases tested with the measured outcome using the script called "*prog*".

Dip Mag-nitude (%)	Dip Duration (cycles)	Dip Phase Offset (degrees)	Detected Dip Type	Comments
91	3	none	0	9% dip undetected.
100	0.9	none	0	Dip too short for recording.
100	2	0	Y	Correct indication.
100	1	180	X	Phase offset causes this to be seen as 2 cycles at 50% - an X.
87	2	0	Y	Correct indication.
87	1.5	180	0	Phase causes this to average into the no-dip area because of short duration.
87	3	180	Y	Extend the dip and we detect it correctly.
58	1.5	0	X	Correct indication.
78	1.5	0	Y	Short duration dip – this averages into Y area.
78	2	0	X	Longer duration dip – this was detected properly.

Fig. 6.5. The Vectograph that was used for test comparison and as a reference of the readings during this research.

6.5. Testing on the prototype board

The harmonic calculations of the prototype have been compared with measurements made with a HP3582A Spectrum Analyser from Hewlett Packard, and a VectoGraph from CT Lab. The Vectograph is an instrument designed to record voltage quality parameters. A photo of the Vectograph is shown in figure 6.5. This can be connected to 3 or 4 wire networks and consists of 5 different, independent recorders in one instrument:

- Digital oscilloscope.
- Spike recorder.
- RMS based event recorder including dips and surges.
- Average profile recorder for harmonic components, THD and phase imbalance.

Dip and surge waveforms were also generated by the simulation programs described in chapter 4 [Ch.4.8.1, p. 87], and the calculated results agreed within 0.5% resolution with signals viewed on a digital storage oscilloscope. It should be noted that these tests were performed with a signal generator, solely for the purpose of verifying actual performance with theory, and do not constitute a specification.

6.6. Testing the communications setup

The information provided in this section pertains to the FieldComm DNP3.0 Test Set which was used to test the protocol and communications interface on the equipment. Details of the communications protocols are already discussed in chapter 3, and the reader is referred to Chapter 3, page 46-60 for complete descriptions of the communications protocol.

6.6.1. The FieldComm test set

The primary use of this software is as a test tool when installing and commissioning DNP3.0 devices. It was also used for capturing DNP3.0 network traffic and analysing the protocol content during this research, using standard PC communications ports and any additional communications devices external to the test set. It can be used in one of two modes:

- **Monitor Mode:** A monitoring session was started to capture and monitor DNP3.0 messages on the communications link. Message details were presented graphically to simplify protocol analysis. Two message displays were provided:

- The Frame View presented the protocol dialog from the viewpoint of the protocol transport functions.
- The Object View presented only the data being exchanged between devices.

- **Master Mode**: FieldComm was also used to simulate a DNP3.0 Level 3 master device by transmitting messages and storing the state of data received from the research RTU. DNP3.0 remote slave devices can also be controlled and monitored in this mode.

6.7. RTU and ENMAC start-up sequence test results

These results were obtained using the FiledComm program to determine if the configuration of the RTU as well as DNP-03 protocol at both the Master and RTU side were satisfactory. These tests were crucial to the complete research project, because at this time the DNP-03 was pioneered for the first time in this configuration at Eskom, and it was important to know if the configuration was successful, and if the Master and RTU successfully communicate with each other via the new protocol. To explain the start-up sequence, it is necessary that for each action a snapshot on FiledComm of the actual event is shown as in figures 6.6 – 6.16.

6.7.1. RTU to Master initial request test results

The author observed, as shown in a snapshot on figure 6.6, that the actual start-up sequence was initiated from Frame packet # 1, where no response was initiated until Frame packet # 17. This was the final request from the RTU to the Master informing the Master it is powered-up and functional.

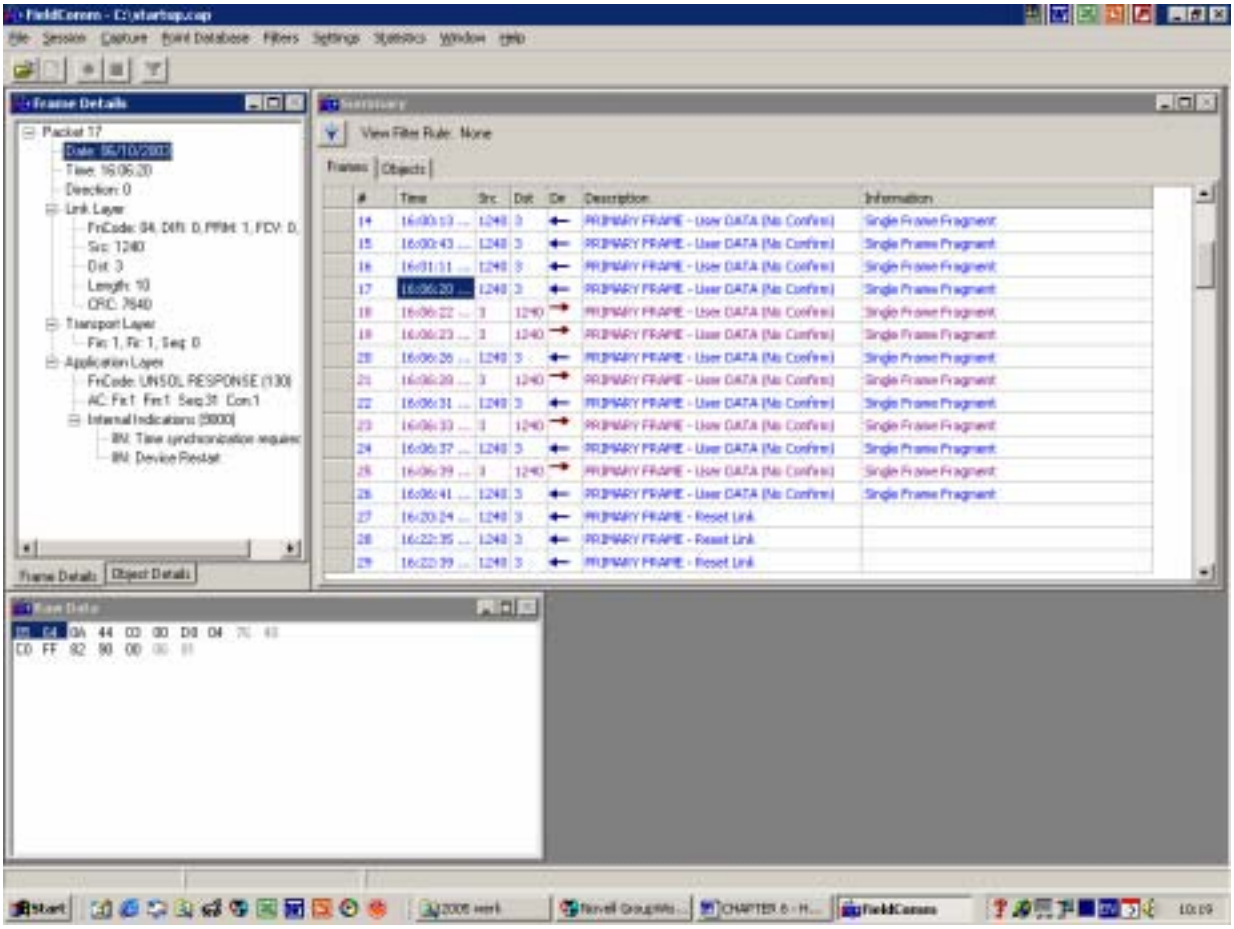


Fig. 6.6. Application layer message from RTU to Master requesting time synch.

During the Frame packet message in line 1-17, at the application layer this field shows the unsolicited responses from the research RTU to the Master station. It informs the Master that the RTU device has restarted, and therefore needs the current date and time from the Master. It also requires Confirm (Con: 1) from the Master. In figure 6.6 it can be seen that the command was repeated from # 1 to # 17, waiting for the Master to confirm. The direction of communication (←) shows the RTU request to the Master. It can be visualised that the Master only confirmed after the 17th request due to other RTU activities for the period of this test. These activities could be:

- Other RTU's were communicating on this channel at this time.
- On this channel, the master was set to queue 22 stations in a sequence.
- The Master first completed the current cycle before responding to the new request.
- The Transmit signal may not appear at the Master (which was not the case here).

On the message in line 18, the application layer shows the direction of communication (→). This shows that the Master (→) now responded to the RTU request, as shown in figures 6.6 - 6.8.

6.7.2. Master to RTU confirmation test results

In figure 6.7 it can be seen that the actual start-up sequence that was initiated from Frame packet # 17, where now confirmed by the Master. In the Application layer, Frame packet # 18, the Master confirms that it has received the unsolicited response from the RTU (as requested in line 17) by issuing a CONFIRM message in the Application layer.

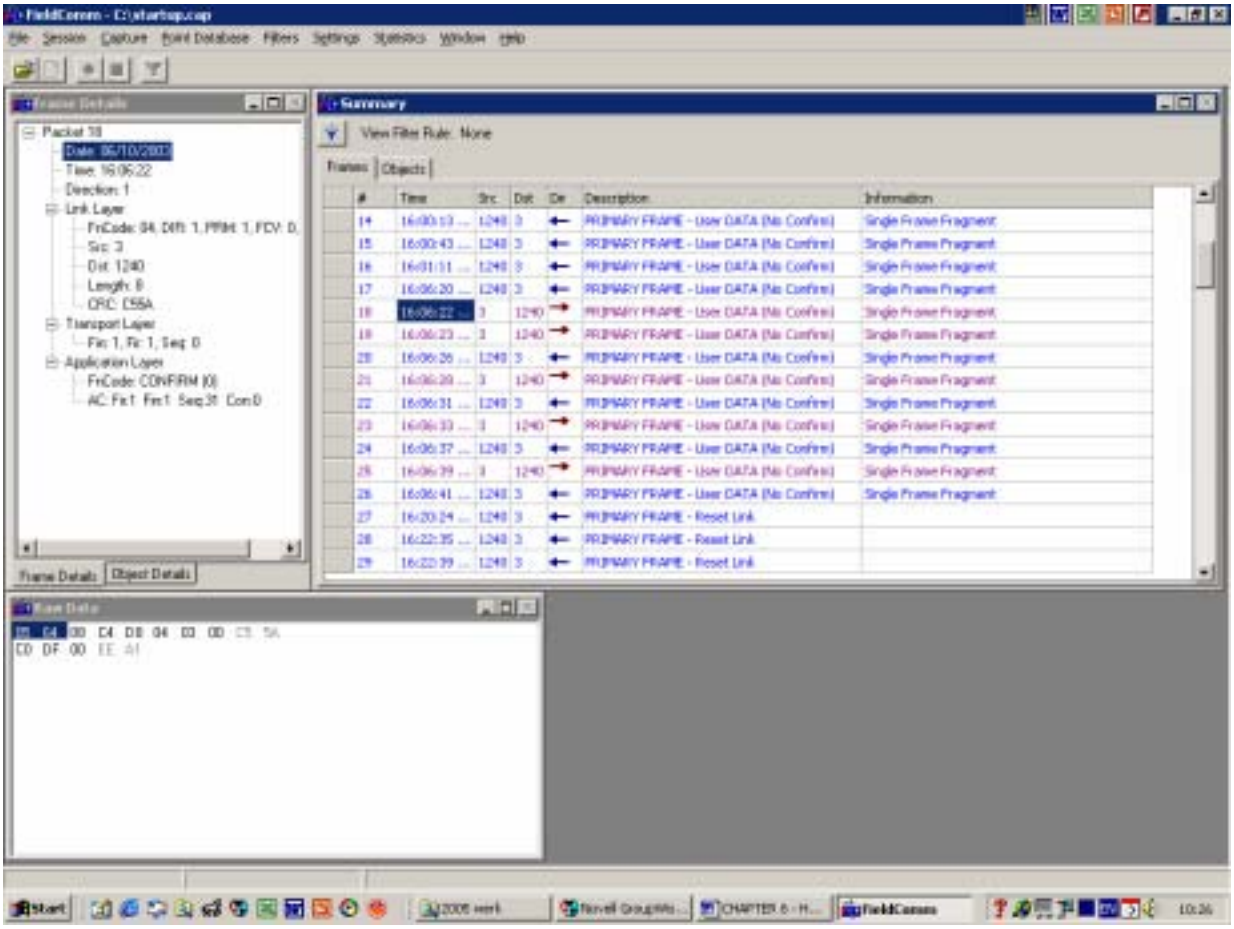


Fig. 6.7. Application layer message of the Master response to the RTU request.

6.7.3. Master response to RTU request test results

In figure 6.8 it can be seen in the Application Layer Frame packet details window on Frame packet # 19 that the Master writes all indications to 0, and clears the Unsolicited Response restart request from Frame packet # 17. The direction of communication (→) shows that it is the Master's response to the RTU's request being processed.

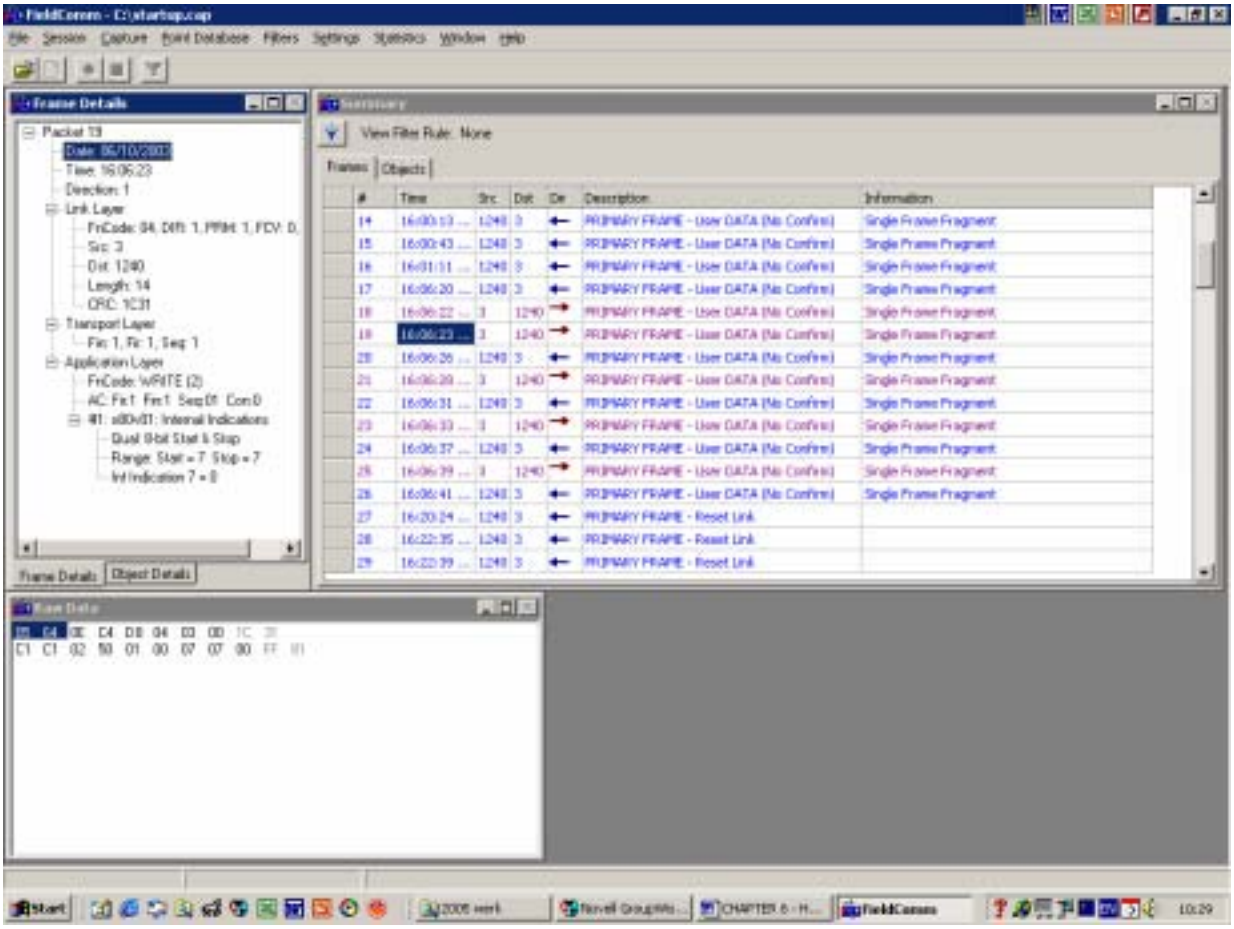


Fig. 6.8. Application layer frame packet details window on Frame packet # 19.

6.7.4. RTU to Master synchronisation request test results

In figure 6.9 it can be seen in the Application Layer Frame packet details window on Frame packet # 20 that the RTU successfully received the "clear restart" from the Master (in 19).

Also in this Frame packet the response from the RTU shows that it now only needs the time synchronisation from the Master to be set at the RTU.

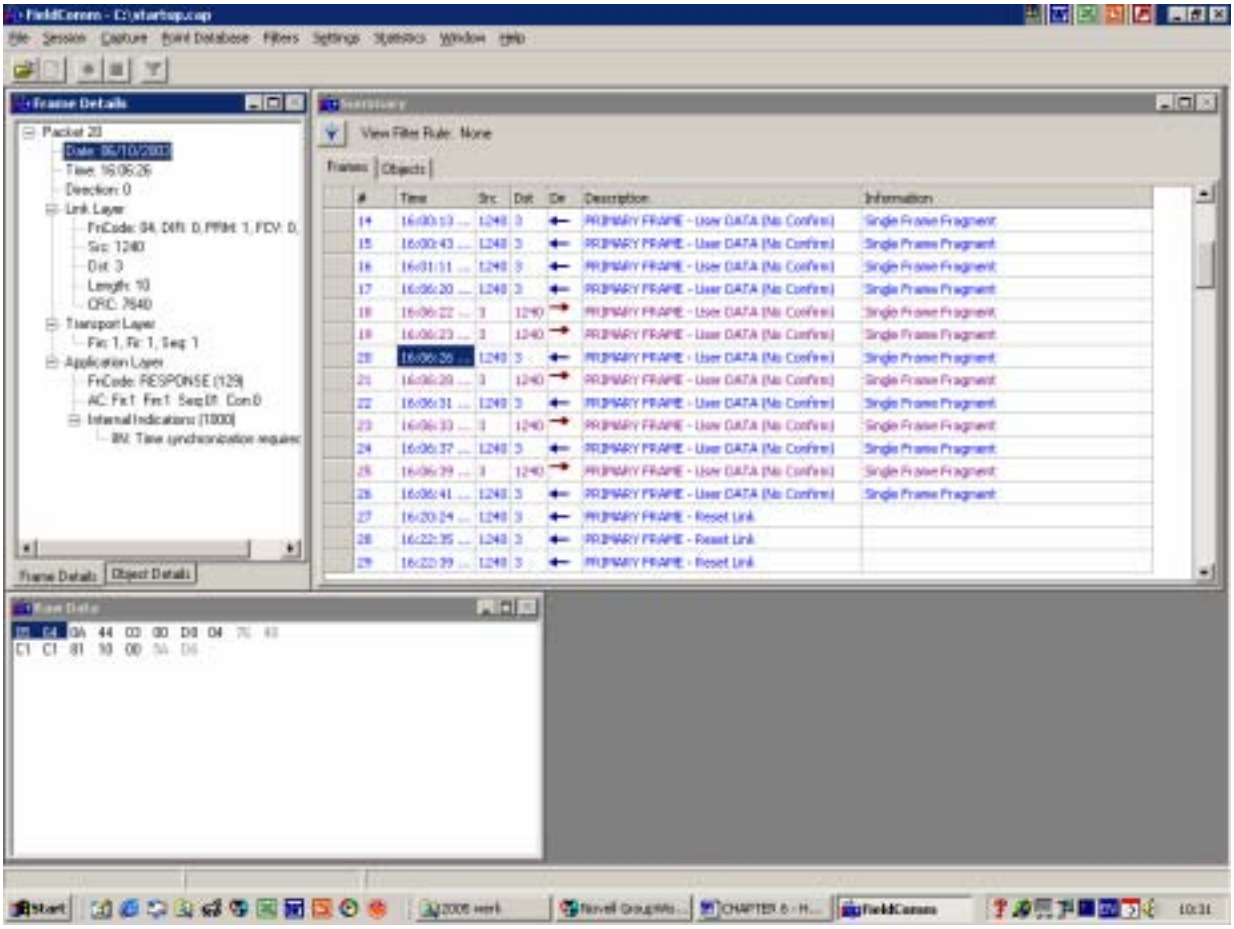


Fig. 6.9. Application layer frame packet details window on Frame packet # 20.

6.7.5. Master to RTU synchronisation set test results

In figure 6.10 it is shown on Frame packet # 21 that the RTU successfully received the synchronisation time and date from the Master. The Sequence-of-event time tagged data (SOE) can now be used with the operation of any recorded dip's, harmonic measurements as well as circuit breakers operations on the Research RTU. Complying with this feature, the Master station will benefit from having data accurately time tagged to allow the analysis of

equipment operations especially during fault finding and for determining the location of the fault and fault conditions.

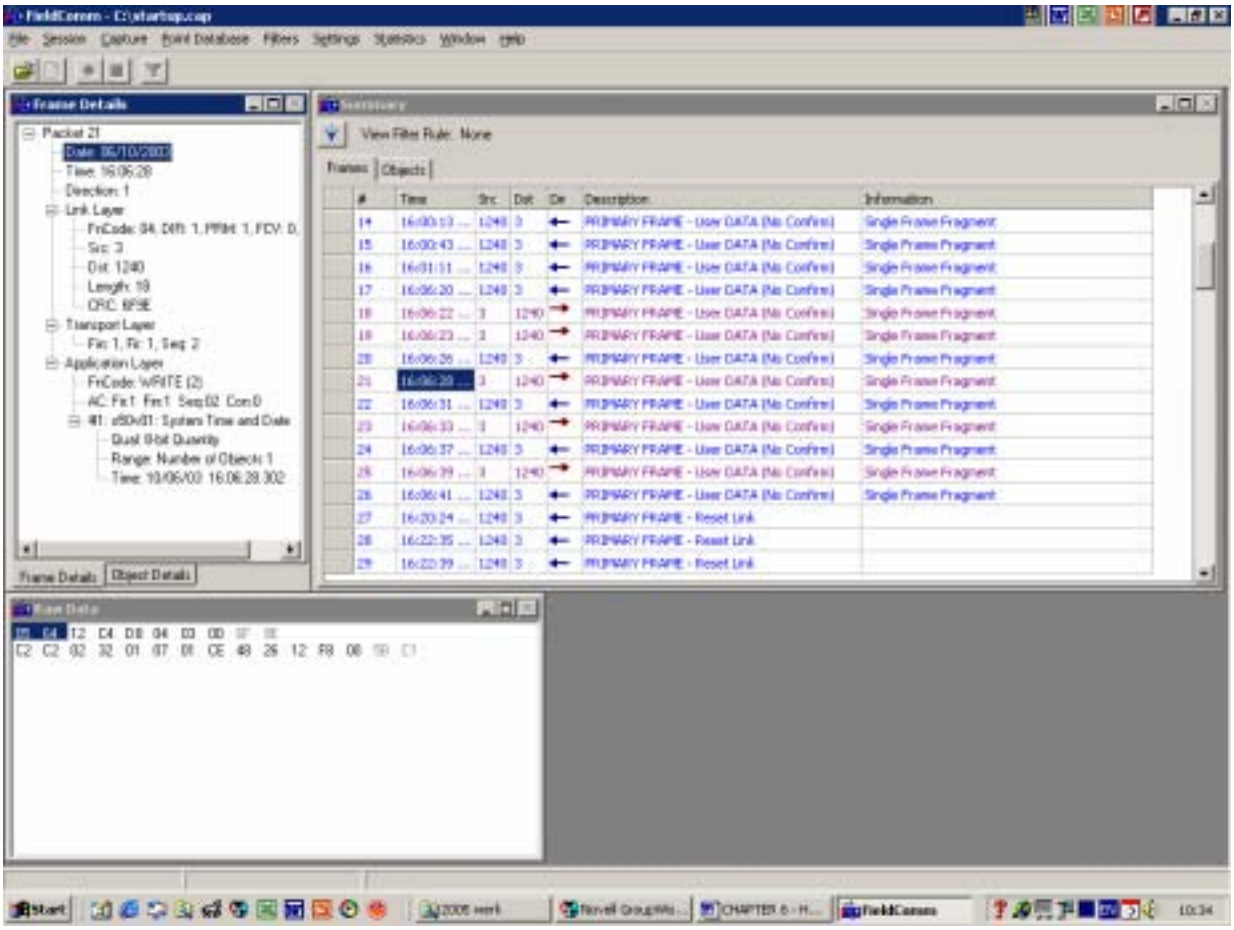


Fig. 6.10. Application layer frame packet details window on Frame packet # 21, time synchronisation setting from Master.

6.7.6. Master to RTU confirmation test results

In figure 6.11 it can be visualised in the Application Layer Frame packet details window on Frame packet # 22 in the Application layer that the RTU responds to the Master's message (time stamping message received in Frame packet # 21), indicating that the RTU now

successfully received the time and date in 21. The direction of communication (←) shows that it is the RTU's response to the Masters previous command.

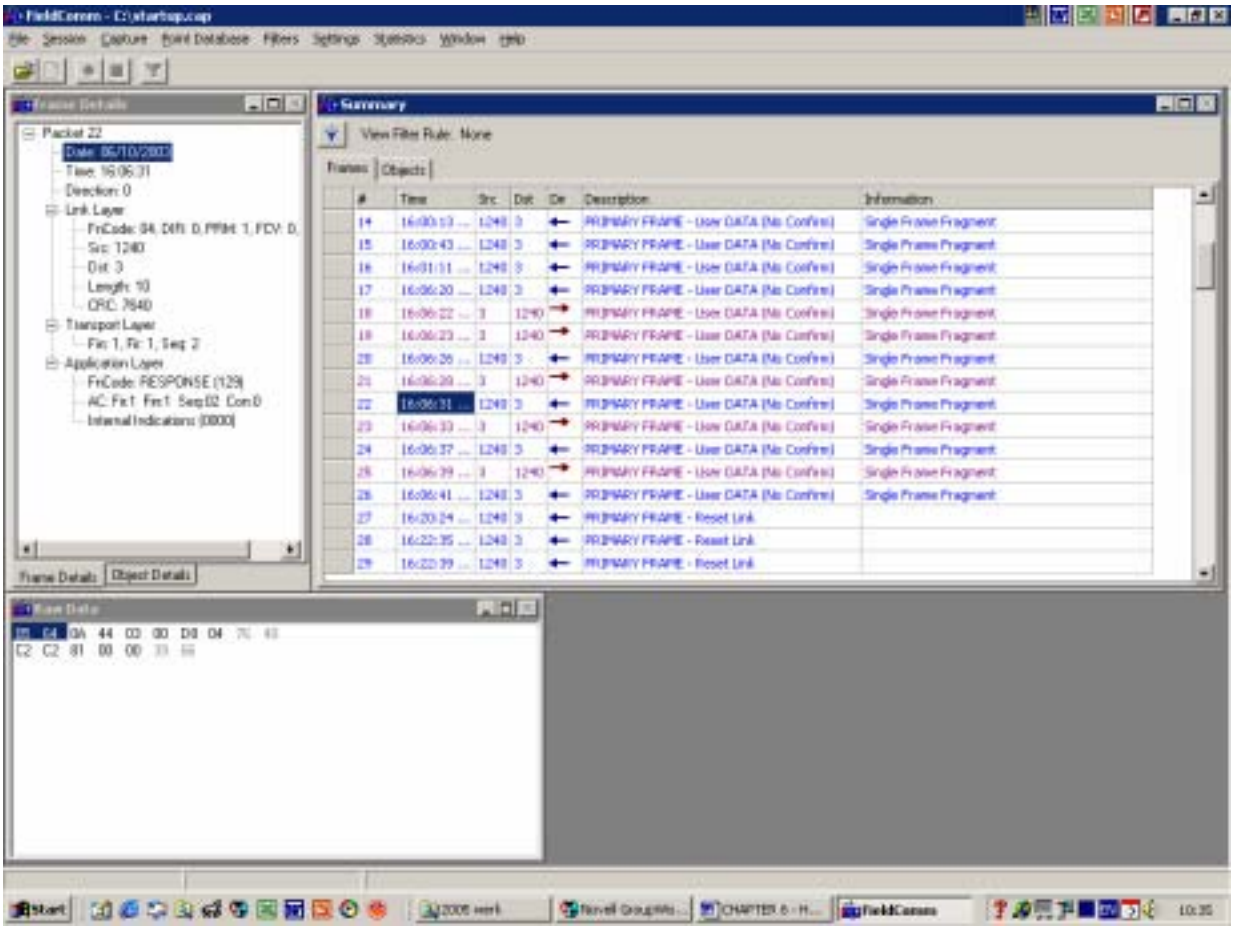


Fig. 6.11. Application layer frame packet details window on Frame packet # 22, time synchronisation response from RTU.

6.7.7. Master to RTU Class 0 – 3 database readings test results

In figure 6.12 the Master now reads the total RTU database from the RTU. In the Application layer Frame, packet # 23, the Master reads all Class 0 – Class 3 data as explained in section 3.6.2. By associating different change event data with different classes, the classes were now

requested with varying periodic rates. An integrity poll, consisting of a class 0 scan, as well as Class 1, 2 and 3 data were performed correctly.

Class 0 were done finally to acquire data not associated with either class 1, 2, or 3. A class 1 poll would ideally be performed as often as possible, while a class 2 poll would be performed less often, and class 3 polls would be performed even less often. For each class data response, only the class data that has changed were returned to keep the response messages small and efficient.

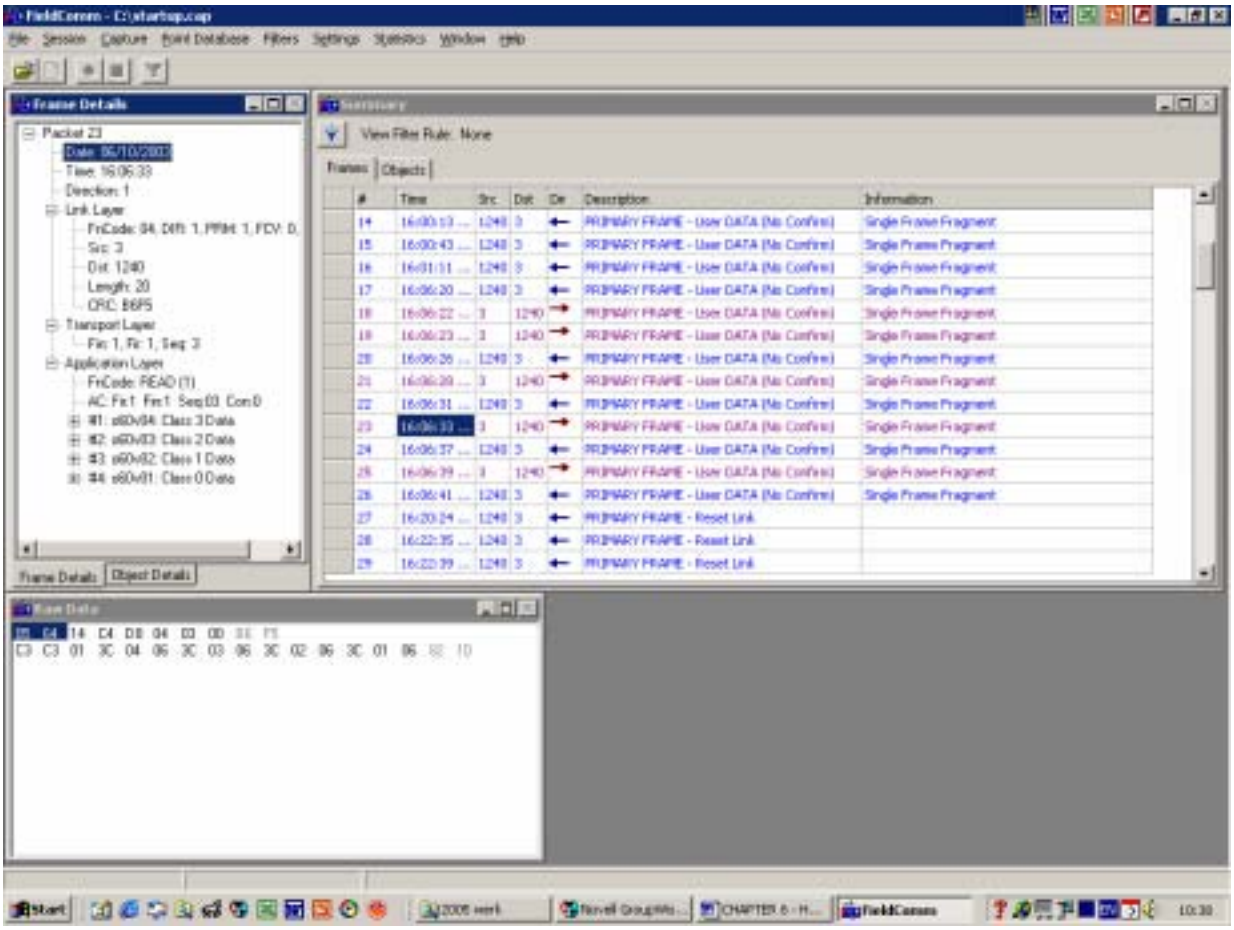


Fig. 6.12. Master reads the total RTU database Class 0 – 3 data from the RTU.

6.7.8. RTU responds to Master with complete database test results

In line 24 the RTU responds to the Master with its total database as configured on UNICON. This includes the digital inputs, analogue inputs etc. This can be viewed in figure 6.13.

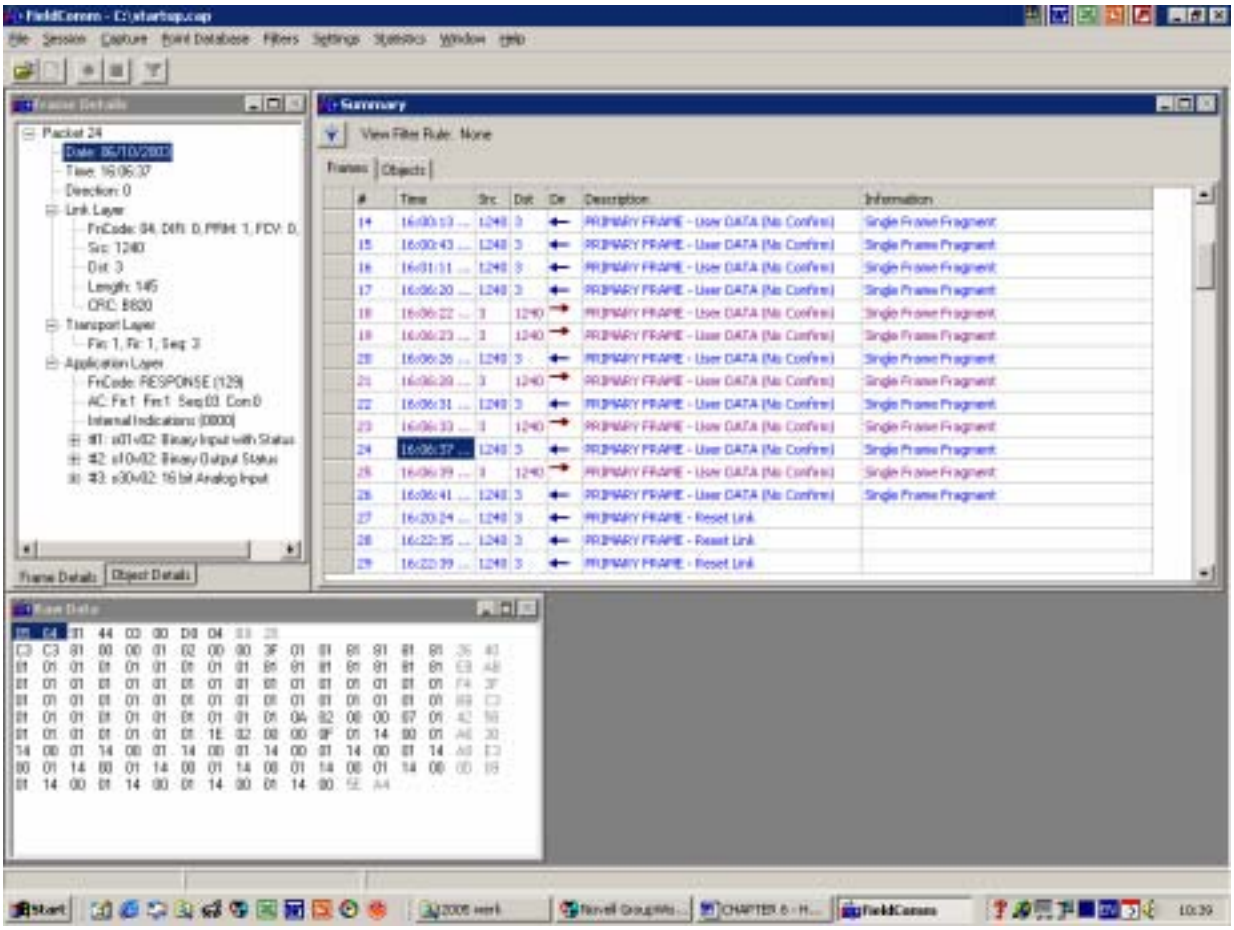


Fig. 6.13. Application layer frame packet details window on Frame packet # 24, RTU response to Master with complete database.

6.7.9. Master enables unsolicited response at RTU test results

In Frame packet # 25 on figure 6.14 the Master now sends a command to the RTU to enable unsolicited responses within Class 1 – Class 3 data.

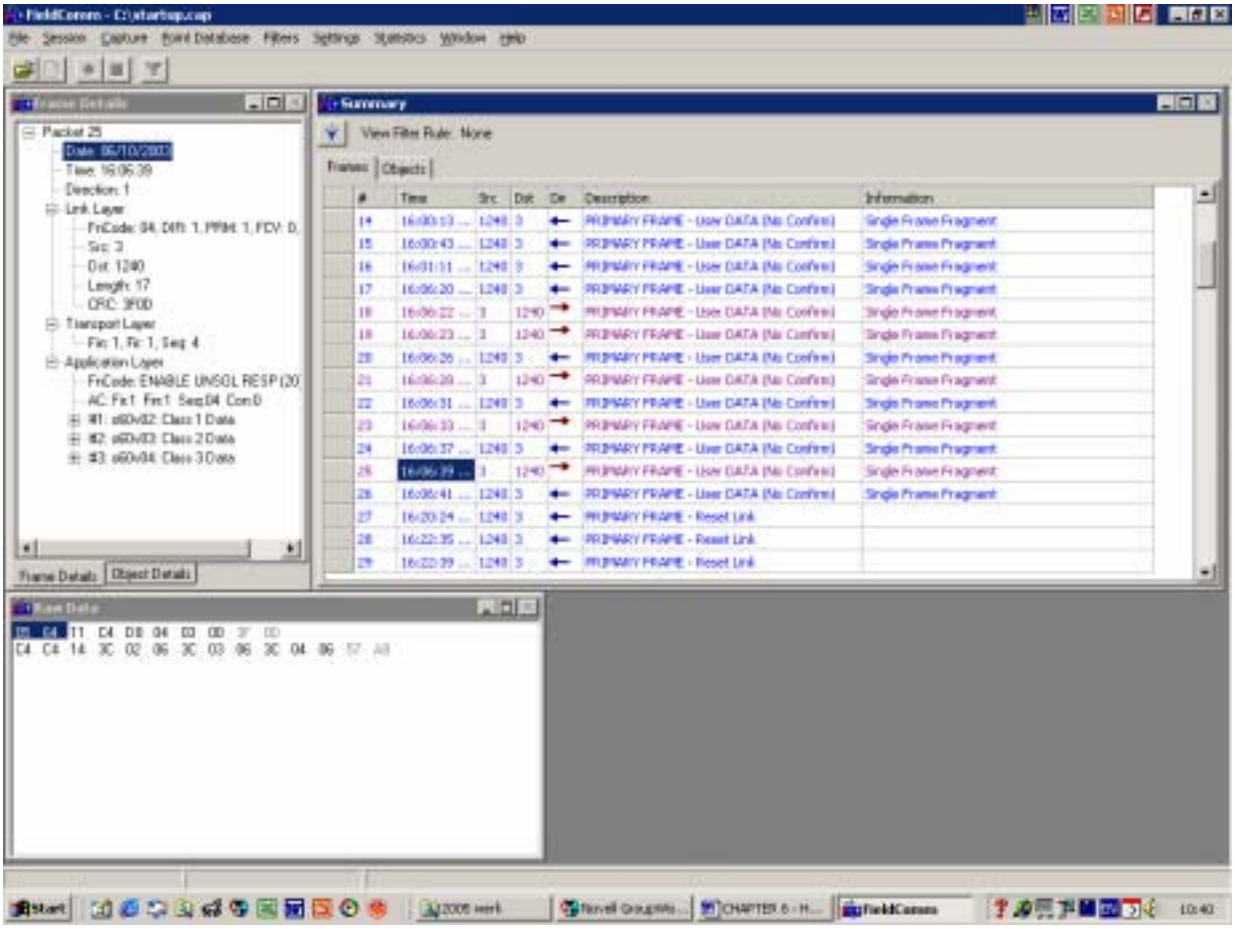


Fig. 6.14. Application layer frame packet details window on Frame packet # 25, Master send unsolicited response to RTU.

6.7.10. RTU response to Master unsolicited request test results

In Frame packet # 26 the RTU response on the Master request (25) can be visualised indicating that the RTU is now in unsolicited mode (figure 6.15). This means that if any event occurs at the RTU, it will be reported to the master immediately.

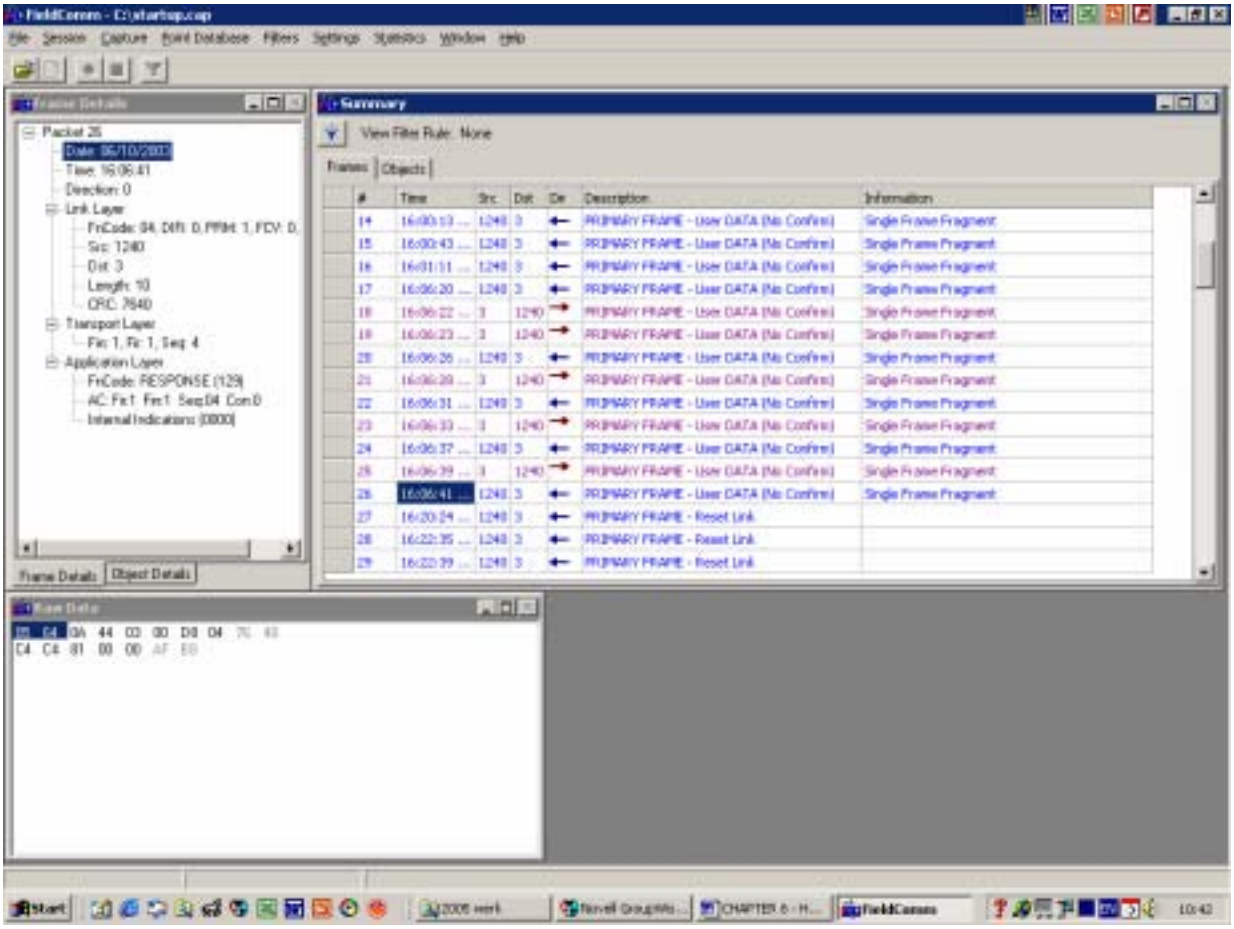


Fig. 6.15. Application layer frame packet details window on Frame packet # 26 showing the RTU's response to the Master unsolicited request.

With the tests performed until now it was clear that the DNP-03 start-up and synchronisation routine was successfully installed and configured, and the RTU was ready to communicate to the Master. The IED was then installed at the RTU, and laboratory tests were carried out on the complete remote system, the RTU and IED, including the harmonic and voltage dip tests, as explained in section 6.4 and 6.5.

6.8. RTU and ENMAC test results on alarm initiation

In this section, only a brief description of the message exchange on the IED digital input changes at the RTU to the Master station are given, as all message changes are exactly the same, with only changes in the digital input or analogue input numbers.

6.8.1. Testing the IED digital input changes

In Frame packet # 102 the RTU sends an unsolicited response to the Master that there was a binary input change (Figure 6.16). In the Application layer it can be seen that the change that occurred was digital input 3 has changed from 0 to 1 (Point 3=1)

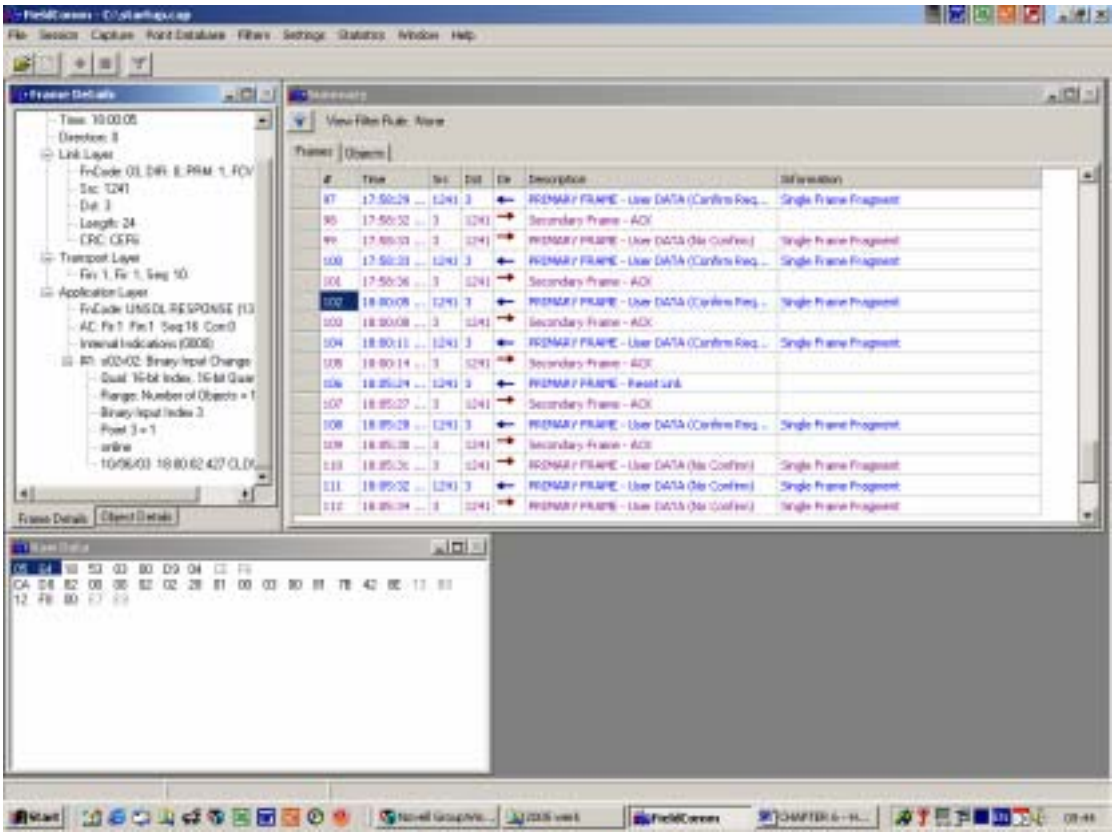


Fig. 6.16. RTU sends an unsolicited response to the Master - Binary input no 3 changes from 0 to 1.

In figure 6.17 below it can be seen in Frame packet # 103 that the Master acknowledges the change. This means that the RTU will stop further communications until the next event occurs, or until it is polled by the Master.

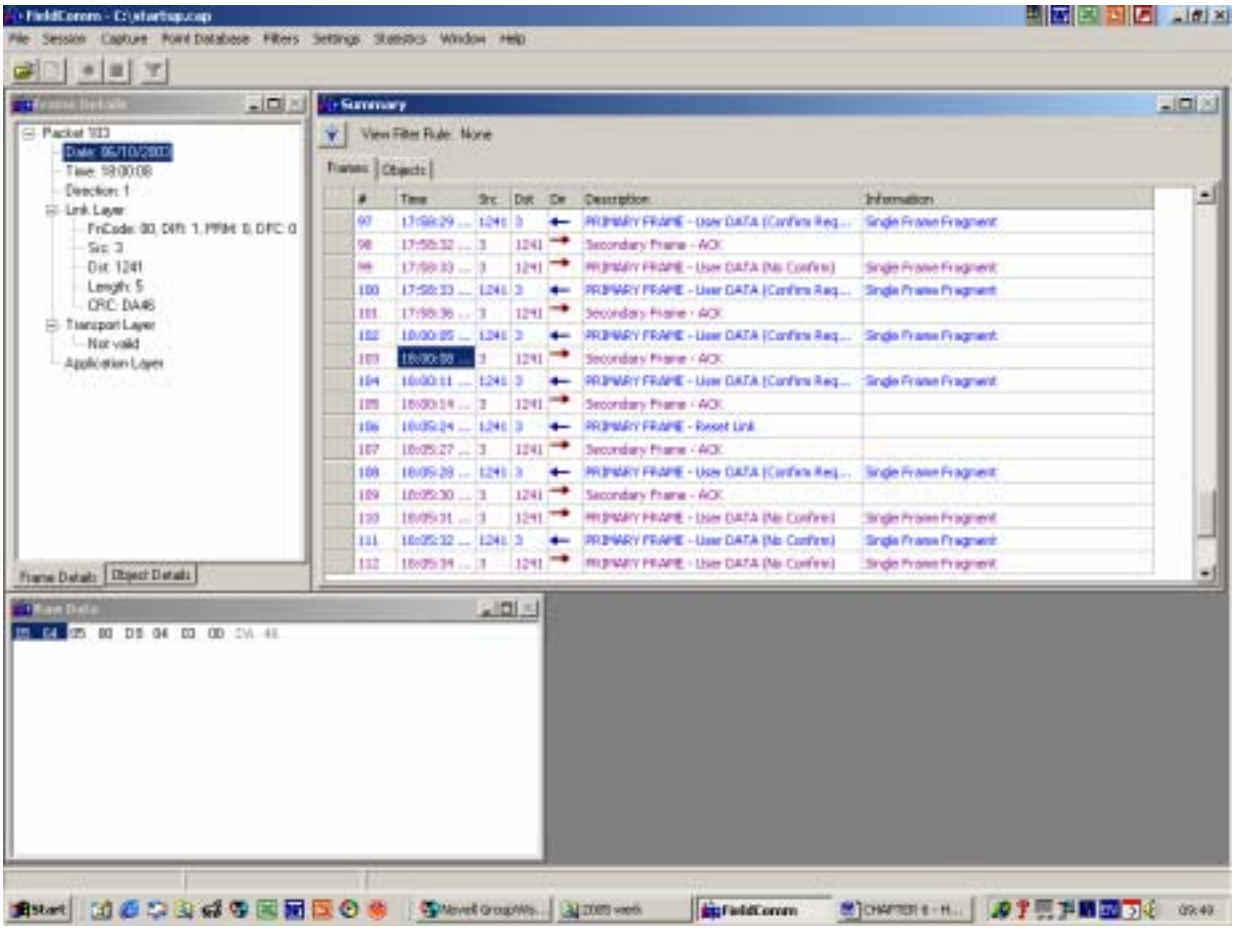


Fig. 6.17. Master acknowledgement of digital input change.

In Frame packet # 104 the RTU sends an unsolicited response to the Master that there was a Binary input change (this time from 1 to 0, in other words the alarm state clears as shown by figure 6.18). In the Application layer it can be visualised that the change that occurred was Digital input 3 has changed back from 1 (alarm state) to 0 (normal state).

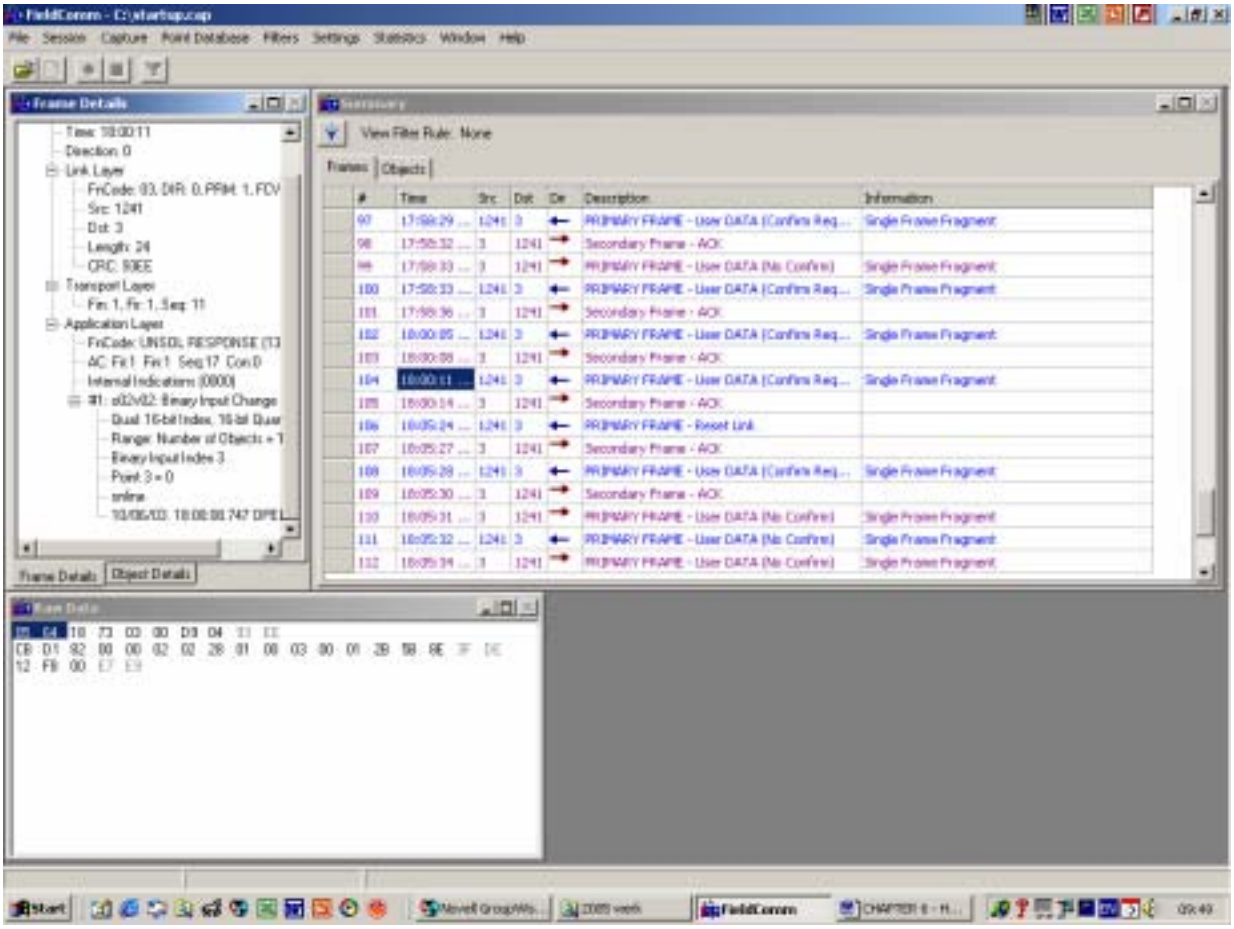


Fig. 6.18. RTU sends an unsolicited response to the Master - Binary input no 3 changes from 1 to 0.

In Frame packet # 105 the Master acknowledges the change (same as 103), and no activities took place until further state changes occurred at the RTU.

This completes the tests on the RTU and ENMAC DNP-03 and IED configuration settings, and the results proved that this part of the research was configured and commissioned successfully. The following tests results were done to prove the IED functionality by simulating harmonics and dips, measuring the results at ENMAC as well as by using test equipment.

6.9. Test results of the harmonic measurements on the IED to ENMAC

All the alarms that were generated by the IED were displayed at the ENMAC. These results were stored in an alarm message file for display. Figure 6.29 shows an extract of an ENMAC alarm message page, showing the alarm changing from normal to low condition on the IED.

6.9.1. Evaluation tests carried out with a signal generator, sine wave

The following tests were carried out in the laboratory using a signal generator, generating a sine wave as a source on the IED. Measurement equipment used during this test:

HP = Hewlett Packard Spectrum Analyzer 3582A analogue analyzer.
VG = VectoGraph measuring system.
IED = IED Development on ENMAC.

Table 6.18 shows the actual measurements taken by using a signal generator. These measurements are compared and shown in figure 6.19, the actual EMNAC screen information captured. These measurements are all relevant to figure 6.22 and 6.25. On the ENMAC screen display, the reader can see that the reading on the ENMAC operator Workstations shows an exact replica of the HP measurements, rounded off without showing the decimal.

Table 6.8. Measurements taken by using a signal generator, sine wave.

GENERATED SIGNAL			MEASURED SIGNAL		
	Frequency	Amplitude	HP Amplitude	VG Amplitude	ENMAC IED Amplitude
Harmonic	Hz	dB	%	%	%
1	50	0	100	100	100
2	100	-35	1.8	1.8	1.86
3	150	-55	0.2	0.2	0.2
4	200	-55	0.2	0.2	0.2
5	250	-68	0	0.1	0
6	300	-65	0.1	0.1	0.1
7	350	-70	0	0	0
8	400	-70	0	0	0
9	450	-65	0.1	0.1	0
10	500	-70	0	0	0
11	550				0
12	600				0
13	650				0
THD %		1.8			1.8

Fig. 6.19. ENMAC screen capture of measurement taken by using a signal generator, sine wave on the IED.

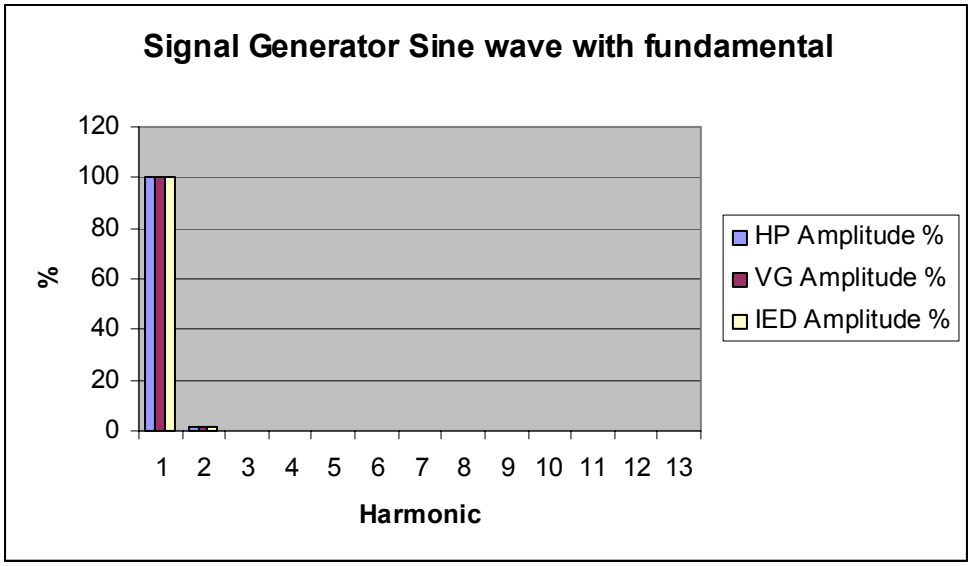


Fig. 6.20. Example of the signal generator, sine wave, including the fundamental.

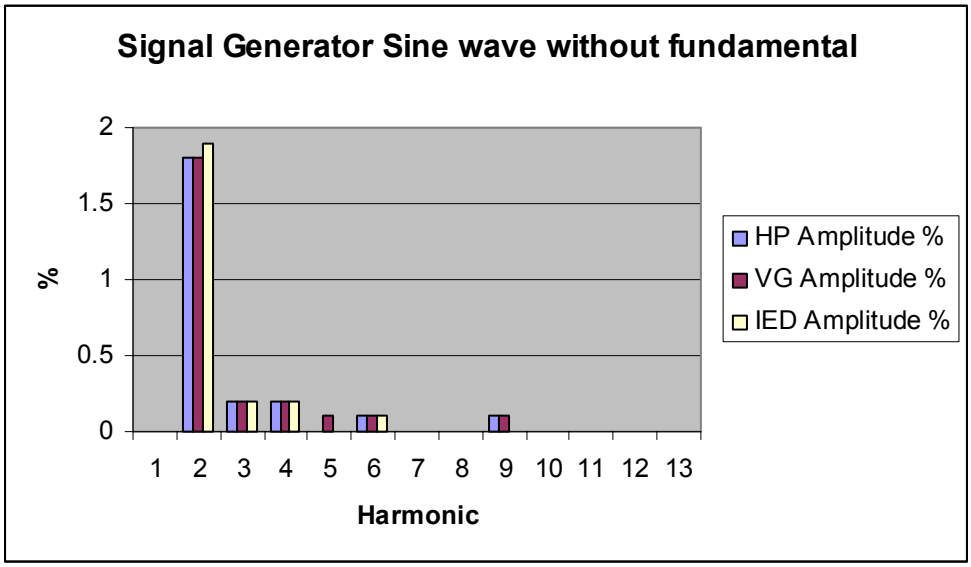


Fig. 6.21. Example of the signal generator, sine wave, without the fundamental.

6.9.2. Evaluation tests carried out with a signal generator, triangular wave

The following tests were carried out in the laboratory using a signal generator, generating a triangular wave as a source on the IED. Measurement equipment used during this test:

HP = Hewlett Packard Spectrum Analyzer 3582A analogue analyzer.

VG = VectoGraph measuring system.

IED = IED Development on ENMAC.

Table 6.9 shows the actual measurements taken by using a signal generator. These measurements are compared to figure 6.22, the actual ENMAC screen information captured. These measurements are all relevant to figure 6.22 and 6.25. On the ENMAC screen display, the reader can see that the reading on the ENMAC operator Workstations shows an exact replica of the HP measurements, rounded of without showing the decimal.

Table 6.9. Measurements taken by using a signal generator, triangular wave.

GENERATED SIGNAL				MEASURED SIGNAL		
						ENMAC
			HP		VG	IED
		Amplitude	Amplitude	Amplitude	Amplitude	Amplitude
Harmonic	Hz	dB	%	dB	%	%
1	50	0	100	0	100	100
2	100	-32	2.5	-32	2.5	2.30
3	150	-20	10	-20	10	9.80
4	200	-39	1.1	-40	1	1.10
5	250	-30	3.2	-30	3.2	3.20
6	300	-44	0.6	-44	0.6	0.60
7	350	-37	1.4	-37	1.4	1.40
8	400	-48	0.4	-48	0.4	0.40
9	450	-43	0.7	-43	0.7	0.70
10	500	-51	0.3	-50	0.3	0.30
11	550	-48	0.4	-48	0.4	0.40
12	600	-54	0.2	-53	0.2	0.20
13	650	-53	0.2	-53	0.2	0.20
THD %		10.7		10.8		10.70

Fundamental	100.00%				
2nd Harm	2.30%	7th Harm	1.40%	12th Harm	0.20%
3rd Harm	9.80%	8th Harm	0.40%	13th Harm	0.20%
4th Harm	1.10%	9th Harm	0.70%	THD	10.70%
5th Harm	3.20%	10th Harm	0.30%	RMS	100.00%
6th Harm	0.60%	11th Harm	0.40%		

Fig. 6.22. ENMAC screen capture of measurement taken by using a signal generator, triangular wave, on the IED.

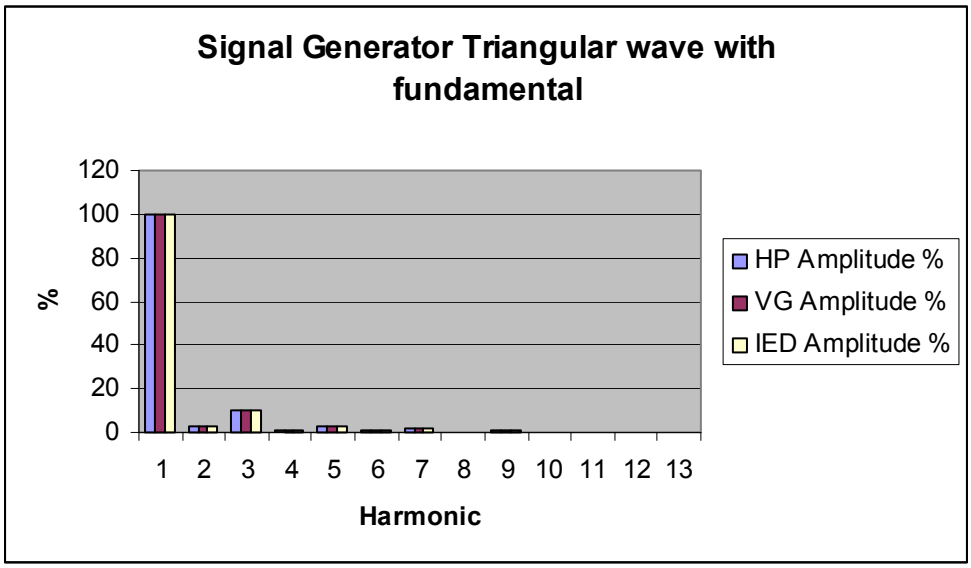


Fig. 6.23. Example of the signal generator, triangular wave, including the fundamental.

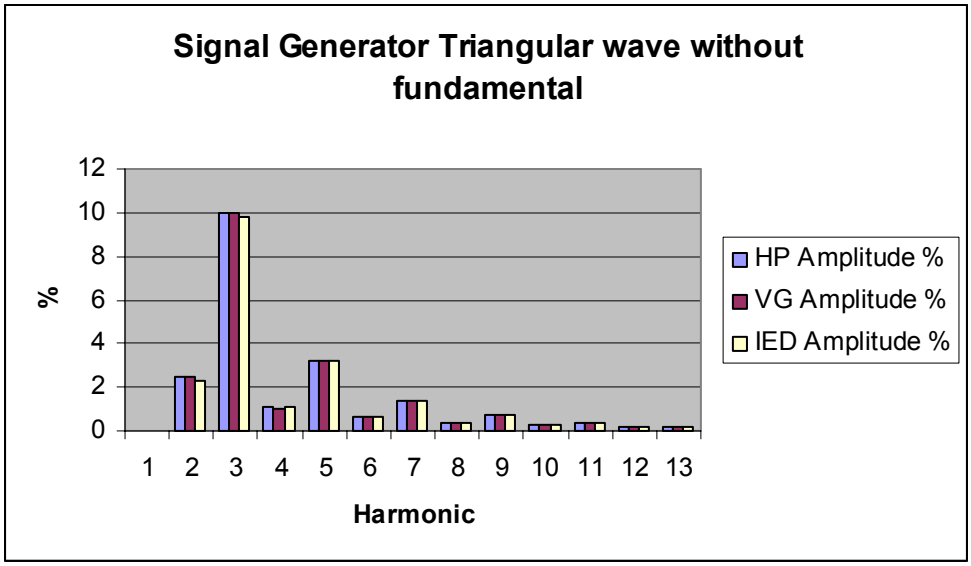


Fig. 6.24. Example of the signal generator, triangular wave, excluding the fundamental.

6.9.3. Evaluation tests carried out with mains signal, sine wave

The following tests were carried out in the laboratory using the mains signal on the building's UPS.

A The 220/110 VAC transformer was used to interface the mains to the IED. Measurement equipment used during this test includes:

HP = Hewlett Packard Spectrum Analyzer 3582A analogue analyzer.

VG = VectoGraph measuring system.

IED = IED Development on ENMAC.

Table 6.10 shows the actual measurements taken by using the VectoGraph. These measurements are shown in figure 6.25, the actual ENMAC screen information captured.

These measurements are all relevant to figure 6.26 and 6.27. On the ENMAC screen display, the reader can see that the reading on the ENMAC operator Workstations shows an exact replica of the HP measurements, rounded of without showing the decimal.

Table 6.10. Measurements taken with the mains signal, 220VAC/110VAC transformer.

MAINS SIGNAL			MEASURED SIGNAL			
						ENMAC
			HP		VG	IED
		Amplitude	Amplitude	Amplitude	Amplitude	Amplitude
Harmonic	Hz	dB	%	dB	%	%
1	50	0	100	0	100	100
2	100	-74	0	-72	0	0.10
3	150	-33	2.2	-33	2.2	2.30
4	200	-76	0	-73	0	0.11
5	250	-33	2.2	-33	2.2	2.40
6	300	-80	0	-80	0	0.01
7	350	-45	0.6	-46	0.5	0.60
8	400	-80	0	-80	0	0.01
9	450	-56	0.2	-57	0.1	0.20
10	500	-80	0	-80	0	0.01
11	550	-72	0	-75	0	0.03
12	600	-80	0	-80	0	0.01
13	650	-74	0	-73	0	0.01
THD %		3.2		3.3		3.22

Fig. 6.25. ENMAC screen capture of measurement taken by the mains signal, 220VAC/110VAC transformer on the IED.

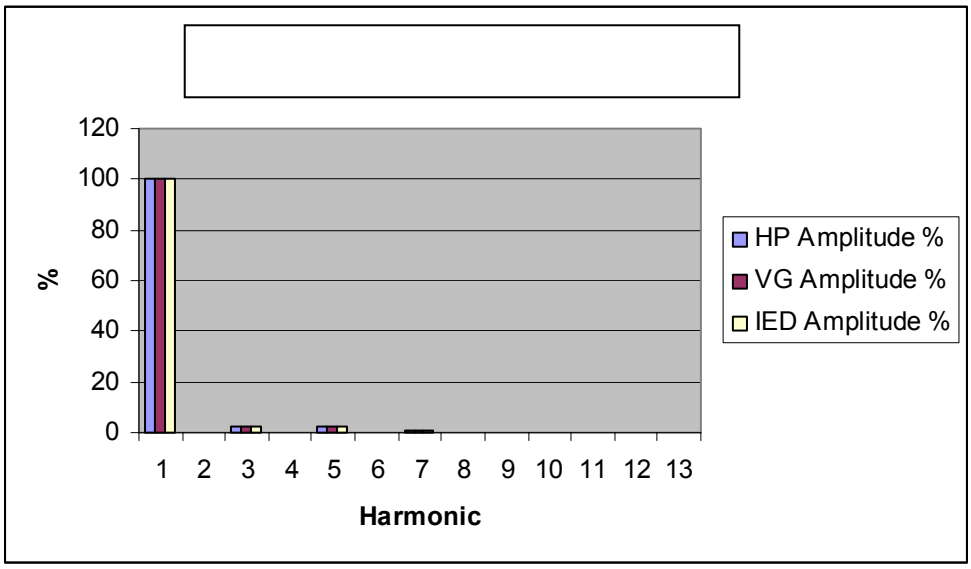


Fig. 6.26. Example of the mains signal, including the fundamental.

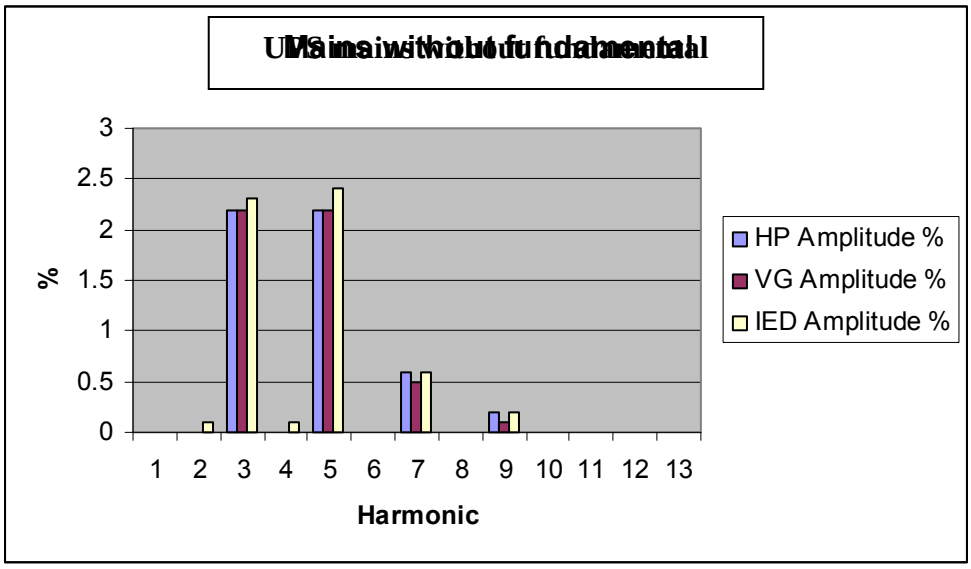


Fig. 6.27. Example of the mains signal, excluding the fundamental.

6.9.3.1. Evaluation tests carried out for the Total Harmonic Distortion measurements

The following results shows the summery of the Total Harmonic Distortion obtained using a signal generator, generating a triangular wave as well as sine wave and also using the mains signal on the IED shown on tables 6.9 & 6.10,

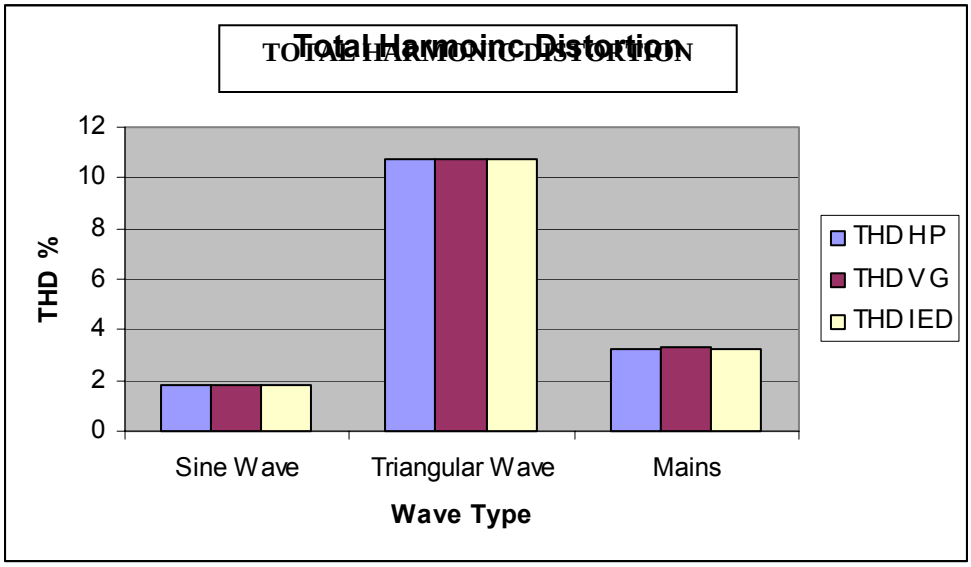


Fig. 6.28. Example of the THD measurements on all the signals.

6.9.4. Observations on the harmonic information as measured on the IED

During this test phase, the following observations were made:

- The first harmonic was always shown as 100.0% irrespective of the actual signal level, and the higher harmonics were relative to the first.
- There was always noise on the signal and from the circuit, and even for no signal input the reading vary one or sometimes two least significant digits. This was avoided on the

final installation by using appropriate earth techniques, as well as screening all input cables.

- In the tests so far, as well as on the final IED installation and other tests, the amplitude shown by the RMS reading accuracy was at all times measured at better than 2%.
- The RMS reading was scaled to 100.0% (where 100% represents that signal whose peak value is half the analogue to digital converter's maximum range). This allows the signal to have a 100% peak over-range for measurements of cycle surges. At 110Volt RMS (the actual output voltage from the VT at all substations) the signal will read 100.0%.
- Furthermore, it was found that if no signal is present, both error flags were set (similar to figure 6.30 as indent 6/4/1/01-03). This was actually not an obstacle, because taking the reading from the IED once managed to clear the flags, so an immediate re-read then do not show the error flags. The flags were set at 1 second intervals.

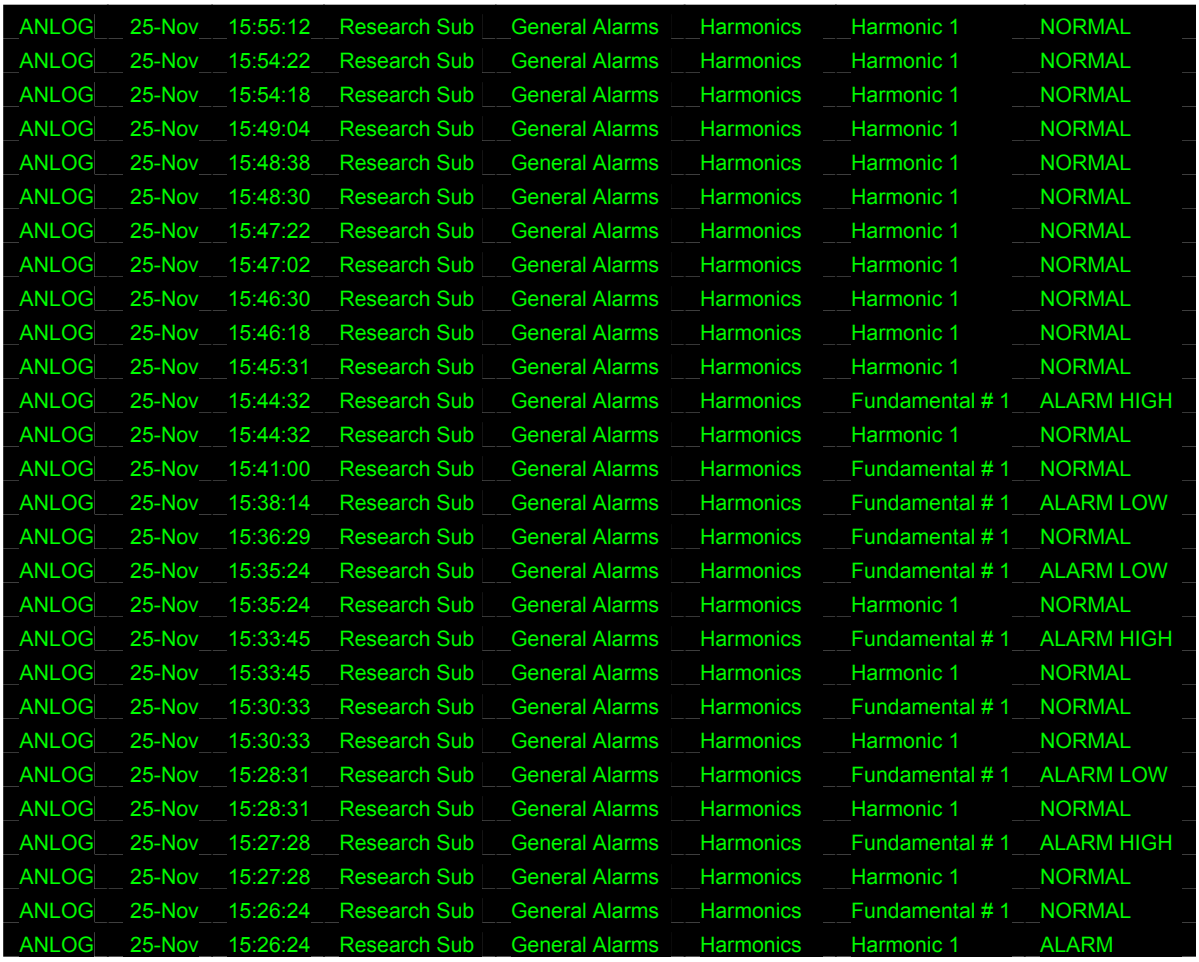
6.9.5. ENMAC Alarms screen capture evaluation

Section 6.4 shows that when an change of state was generated correctly at the RTU, and then received at the ENMAC Master station, for instance if it was a harmonic content change, the following happened.

1) The change of state was forwarded to the NMS server.
2) The value was compared with the GRTV - and if there was a difference,
3) A new alarm was generated on the OW alarm screen, as shown in figure 6.29. This

must be read from bottom to top.

Note: If the value is the same as the GRTV, then the alarm must have been generated already e.g. state "1" (alarm state not changing), so a new alarm is **not** generated - this is to avoid the same alarm being generated more than once (e.g. during a changeover on some systems).



ANLOG	25-Nov	15:55:12	Research Sub	General Alarms	Harmonics	Harmonic 1	NORMAL
ANLOG	25-Nov	15:54:22	Research Sub	General Alarms	Harmonics	Harmonic 1	NORMAL
ANLOG	25-Nov	15:54:18	Research Sub	General Alarms	Harmonics	Harmonic 1	NORMAL
ANLOG	25-Nov	15:49:04	Research Sub	General Alarms	Harmonics	Harmonic 1	NORMAL
ANLOG	25-Nov	15:48:38	Research Sub	General Alarms	Harmonics	Harmonic 1	NORMAL
ANLOG	25-Nov	15:48:30	Research Sub	General Alarms	Harmonics	Harmonic 1	NORMAL
ANLOG	25-Nov	15:47:22	Research Sub	General Alarms	Harmonics	Harmonic 1	NORMAL
ANLOG	25-Nov	15:47:02	Research Sub	General Alarms	Harmonics	Harmonic 1	NORMAL
ANLOG	25-Nov	15:46:30	Research Sub	General Alarms	Harmonics	Harmonic 1	NORMAL
ANLOG	25-Nov	15:46:18	Research Sub	General Alarms	Harmonics	Harmonic 1	NORMAL
ANLOG	25-Nov	15:45:31	Research Sub	General Alarms	Harmonics	Harmonic 1	NORMAL
ANLOG	25-Nov	15:44:32	Research Sub	General Alarms	Harmonics	Fundamental # 1	ALARM HIGH
ANLOG	25-Nov	15:44:32	Research Sub	General Alarms	Harmonics	Harmonic 1	NORMAL
ANLOG	25-Nov	15:41:00	Research Sub	General Alarms	Harmonics	Fundamental # 1	NORMAL
ANLOG	25-Nov	15:38:14	Research Sub	General Alarms	Harmonics	Fundamental # 1	ALARM LOW
ANLOG	25-Nov	15:36:29	Research Sub	General Alarms	Harmonics	Fundamental # 1	NORMAL
ANLOG	25-Nov	15:35:24	Research Sub	General Alarms	Harmonics	Fundamental # 1	ALARM LOW
ANLOG	25-Nov	15:35:24	Research Sub	General Alarms	Harmonics	Harmonic 1	NORMAL
ANLOG	25-Nov	15:33:45	Research Sub	General Alarms	Harmonics	Fundamental # 1	ALARM HIGH
ANLOG	25-Nov	15:33:45	Research Sub	General Alarms	Harmonics	Harmonic 1	NORMAL
ANLOG	25-Nov	15:30:33	Research Sub	General Alarms	Harmonics	Fundamental # 1	NORMAL
ANLOG	25-Nov	15:30:33	Research Sub	General Alarms	Harmonics	Harmonic 1	NORMAL
ANLOG	25-Nov	15:28:31	Research Sub	General Alarms	Harmonics	Fundamental # 1	ALARM LOW
ANLOG	25-Nov	15:28:31	Research Sub	General Alarms	Harmonics	Harmonic 1	NORMAL
ANLOG	25-Nov	15:27:28	Research Sub	General Alarms	Harmonics	Fundamental # 1	ALARM HIGH
ANLOG	25-Nov	15:27:28	Research Sub	General Alarms	Harmonics	Harmonic 1	NORMAL
ANLOG	25-Nov	15:26:24	Research Sub	General Alarms	Harmonics	Fundamental # 1	NORMAL
ANLOG	25-Nov	15:26:24	Research Sub	General Alarms	Harmonics	Harmonic 1	ALARM

Fig. 6.29. The ENMAC alarms screen capturing.

6.10. Dip evaluation – test to Video Terminal

An example of the MODBUS Plant Settings is shown by the VT display in figure 6.30. Line 1-8 shows the 8 x 8 digital inputs of the M/IO and lines 9-14 shows the 8 x 6 digital inputs of the MODBUS IED. This view shows the changes (MSB to LSB) on the 8-bit indications in line 9-11 where the IED was actually not connected to an external source and therefore showing bits 7 and 8 fluctuating.

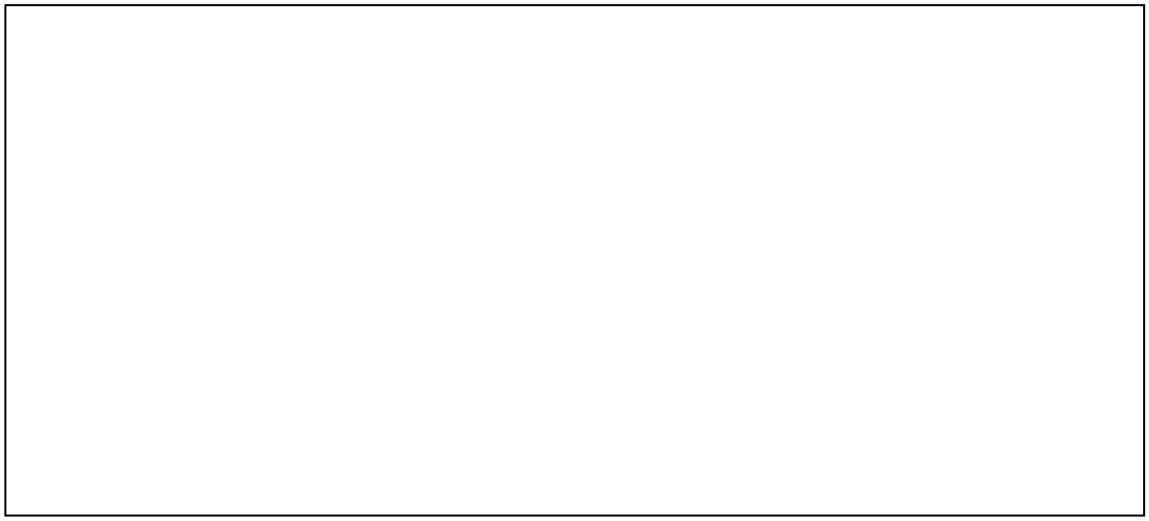


Fig. 6.30. Example of the Video Terminal display captured during the IED tests.

6.11. Voltage dip test results

During this section, a variable 110VAC transformer was initially used to simulate the dips on each phase of certain time durations. Different voltage dips were simulated by switching between the variable transformer voltage and a constant 110VAC reference voltage, using a precision timer. This seemed to make the testing procedure complicated and tended to be a shoddy approach. This method of testing was then abandoned.

The author subsequently used an Omicron CMC 156 three phase voltage / three phase current test set. This test set has a state sequencer option, a flexible tool for determining operating time and logical timing sequences. A state is defined by the output conditions (voltage) and the condition for the ending of the state. Several individual states could be strung together in order to define a test sequence.

The transition from one state to the next could take place after a fixed time or by pressing a key. This was then used for testing, and these tests allowed the author to get the ENMAC results and to compare it with actual readings from the Omicron.

Figure 6.31 shows the ENMAC response at 12h01 and 12h13 on 11/10/2003 for a NRS Class X and Y dip, shown on line 4 of table 6.11. This alarm will flash in the red colour until accepted by the operator.

Table. 6.11. Table of the set of IED alarms captured on ENMAC for test purposes.

Number	Date	Time	Phase	Duration msec.	Omicron %	IED Deviation %	ENMAC IED NRS Alarm
1	11/10/2003	12:01 AM	A	20	-15.1	-15.07%	Y
2	11/10/2003	12:01 AM	B	20	-15.1	-15.08%	Y
3	11/10/2003	12:01 AM	C	30	-15.1	-15.08%	Y
4	**11/10/2003**	**12:01 AM**	**A**	**30**	**-15.2**	**-15.18%**	**Y**
5	11/10/2003	12:13 AM	B	30	-40.6	-40.56%	X
6	11/10/2003	12:49 AM	C	30	-15.5	-15.48%	Y
7	11/10/2003	12:55 AM	A	30	-16.4	-16.38%	Y
8	11/10/2003	10:47 AM	B	20	-27.5	-27.47%	X
9	11/10/2003	11:44 AM	C	30	-15.1	-15.08%	Y

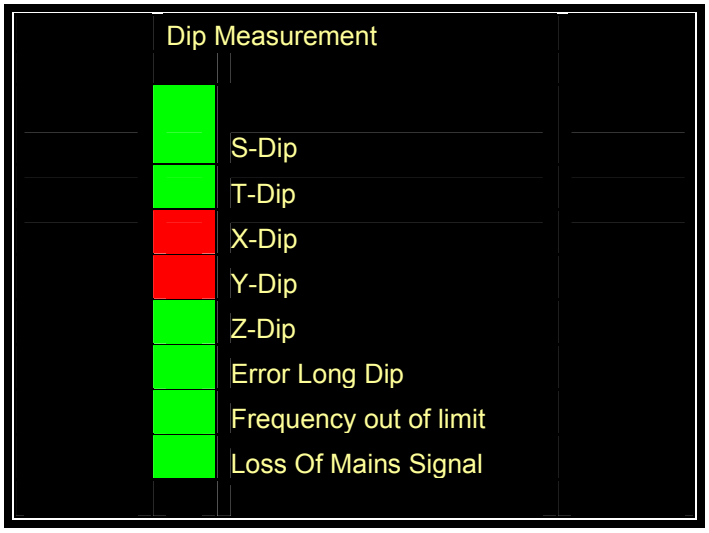


Fig. 6.31. The ENMAC display for the NRS class X and Y-dips on the ENMAC alarm screen.

Figure 6.32 shows the ENMAC response at 12h55 on 12/10/2003 for a NRS class Z dip, shown on line 7 of table 6.12. This alarm will flash in the red colour until accepted by the operator.

Table. 6.12. Table of the second set of IED alarms captured on ENMAC for test purposes.

Number	Date	Time	Phase	Duration msec.	Omicron %	IED Deviation	ENMAC IED NRS Alarm
1	12/10/2003	12:01 AM	A	370	-28.8	-28.77%	S
2	12/10/2003	12:01 AM	B	720	-41.9	-41.86%	Z
3	12/10/2003	12:01 AM	C	3830	-45.9	-45.85%	X
4	12/10/2003	12:13 AM	A	140	-44.8	-44.76%	X
5	12/10/2003	12:34 AM	B	250	-41.1	-41.06%	S
6	12/10/2003	12:54 AM	C	380	-45.4	-45.35%	S
7	**12/10/2003**	**12:55 AM**	**A**	**720**	**-41.9**	**-41.86%**	**Z**
8	12/10/2003	12:58 AM	B	130	-43.9	-43.86%	X
9	12/10/2003	01:13 AM	C	150	-46.1	-46.05%	S
10	12/10/2003	02:13 AM	A	240	-21.2	-21.18%	S
11	12/10/2003	09:01 AM	B	950	-42.1	-42.06%	Z

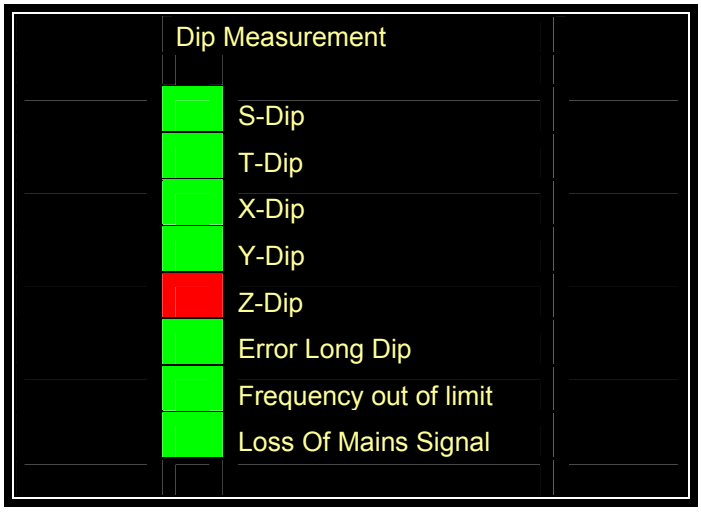


Fig. 6.32. The ENMAC display for the NRS class Z-dip on the ENMAC alarm screen.

Figure 6.33 shows the ENMAC response at 10H47 on 14/10/2003 for a NRS class S dip, shown on line 3 of table 6.13. This alarm will flash in the red colour until accepted by the operator.

Table. 6.13. Table of the third set of IED alarms captured on ENMAC for test purposes.

Number	Date	Time	Phase	Duration msec.	Omicron %	IED Deviation	ENMAC IED NRS Alarm
1	14/10/2003	10:05 AM	A	60	-70	-69.93%	T
2	14/10/2003	10:31 AM	A	1490	-41.8	-41.76%	Z
3	**14/10/2003**	**10:47 AM**	**A**	**440**	**-20.4**	**-20.38%**	**S**
4	14/10/2003	11:01 AM	B	1590	-38.5	-38.46%	Z
5	14/10/2003	11:05 AM	B	60	-70	-69.93%	T
6	14/10/2003	11:47 AM	B	620	-42.1	-42.06%	Z
7	14/10/2003	12:05 PM	C	70	-24	-23.98%	X
8	14/10/2003	12:21 PM	C	1810	-38.7	-38.66%	Z
9	14/10/2003	12:47 PM	C	620	-41.2	-41.16%	Z
10	14/10/2003	01:47 PM	A	700	-41.5	-41.46%	Z

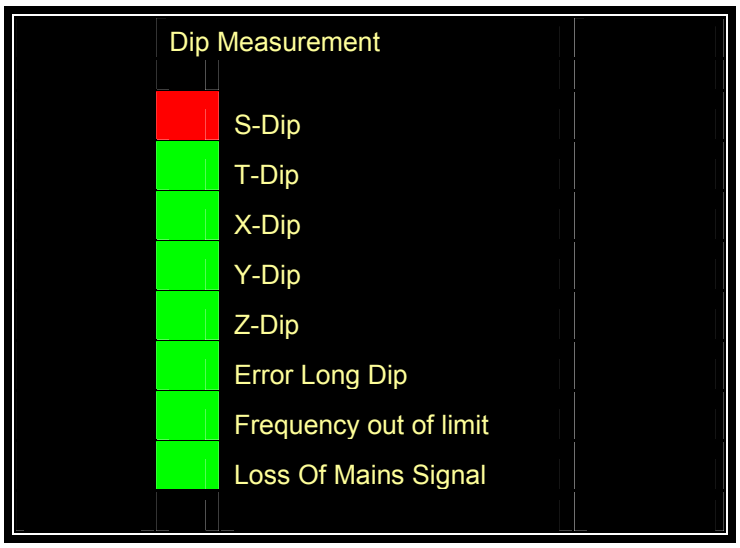


Fig. 6.33. The ENMAC display for the NRS class S-dip on the ENMAC alarm screen.

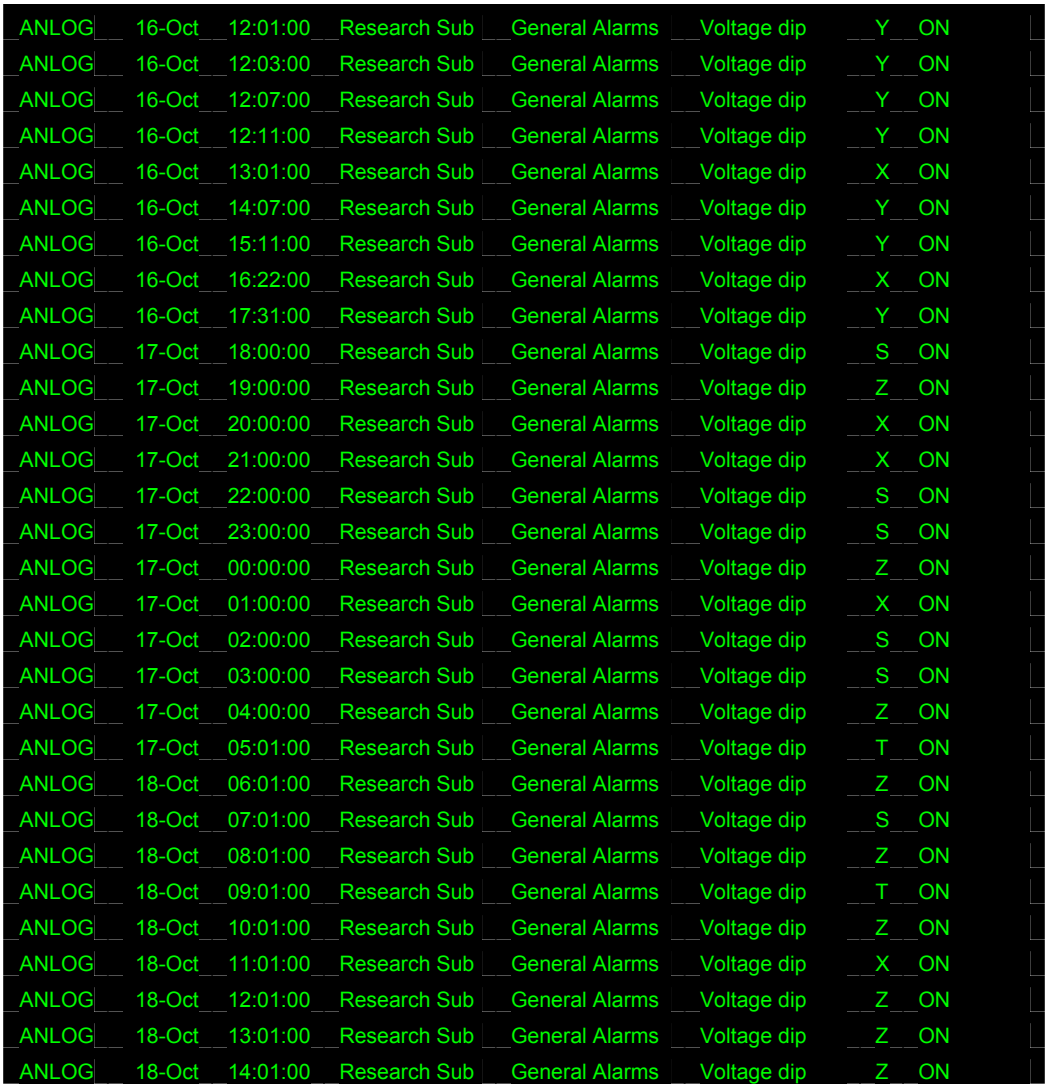
In the next table (Table 6.14), the same sequence was run at a different time and date. Table 6.14 shows an example of some data that was captured during this period of the research. Figure 6.34 shows the ENMAC alarm page, filtered on "Research sub", alarm state "on".

Table 6.14. Table of comparison of voltage dip alarms generated by the Omicron test set.

Number	Date	Time	Phase	Duration msec.	Omicron %	IED Deviation	ENMAC IED NRS Alarm
1	16/10/2003	12:01 PM	A	20	-15.21	-15.21%	Y
2	16/10/2003	12:03 PM	A	20	-15.21	-15.21%	Y
3	16/10/2003	12:07 PM	A	30	-15.21	-15.21%	Y
4	16/10/2003	12:11 PM	A	30	-15.31	-15.31%	Y
5	16/10/2003	01:01 PM	A	30	-40.71	-40.71%	X
6	16/10/2003	02:07 PM	A	30	-15.61	-15.61%	Y
7	16/10/2003	03:11 PM	A	30	-16.51	-16.51%	Y
8	16/10/2003	04:22 PM	A	20	-27.61	-27.61%	X
9	16/10/2003	05:31 PM	A	30	-15.21	-15.21%	Y
10	17/10/2003	06:00 PM	A	370	-28.91	-28.91%	S
11	17/10/2003	07:00 PM	A	720	-42.01	-42.01%	Z
12	17/10/2003	08:00 PM	A	3830	-46.01	-46.01%	X

13	17/10/2003	09:00 PM	A	140	-44.91	-44.91%	X
14	17/10/2003	10:00 PM	A	250	-41.21	-41.21%	S
15	17/10/2003	11:00 PM	A	380	-45.51	-45.51%	S
16	17/10/2003	12:00 AM	A	720	-42.01	-42.01%	Z
17	17/10/2003	01:00 AM	A	130	-44.01	-44.01%	X
18	17/10/2003	02:00 AM	A	150	-46.21	-46.21%	S
19	17/10/2003	03:00 AM	A	240	-21.31	-21.31%	S
20	17/10/2003	04:00 AM	A	950	-42.21	-42.21%	Z
21	17/10/2003	05:01 AM	A	60	-70.11	-70.11%	T
22	18/10/2003	06:01 AM	A	1490	-41.91	-41.91%	Z
23	18/10/2003	07:01 AM	A	440	-20.51	-20.51%	S
24	18/10/2003	08:01 AM	A	1590	-38.61	-38.61%	Z
25	18/10/2003	09:01 AM	A	60	-70.11	-70.11%	T
26	18/10/2003	10:01 AM	A	620	-42.21	-42.21%	Z
27	18/10/2003	11:01 AM	A	70	-24.11	-24.11%	X
28	18/10/2003	12:01 PM	A	1810	-38.81	-38.81%	Z
29	18/10/2003	01:01 PM	A	620	-41.31	-41.31%	Z
30	18/10/2003	02:01 PM	A	700	-41.61	-41.61%	Z

Figure 6.34 shows the ENMAC alarm page for multiple alarms received from the IED at 16 October 2003 for a NRS class dips shown in table 6.14. This alarm will stay on the alarm page until cleared by the IED.



ANLOG	16-Oct	12:01:00	Research Sub	General Alarms	Voltage dip	Y	ON
ANLOG	16-Oct	12:03:00	Research Sub	General Alarms	Voltage dip	Y	ON
ANLOG	16-Oct	12:07:00	Research Sub	General Alarms	Voltage dip	Y	ON
ANLOG	16-Oct	12:11:00	Research Sub	General Alarms	Voltage dip	Y	ON
ANLOG	16-Oct	13:01:00	Research Sub	General Alarms	Voltage dip	X	ON
ANLOG	16-Oct	14:07:00	Research Sub	General Alarms	Voltage dip	Y	ON
ANLOG	16-Oct	15:11:00	Research Sub	General Alarms	Voltage dip	Y	ON
ANLOG	16-Oct	16:22:00	Research Sub	General Alarms	Voltage dip	X	ON
ANLOG	16-Oct	17:31:00	Research Sub	General Alarms	Voltage dip	Y	ON
ANLOG	17-Oct	18:00:00	Research Sub	General Alarms	Voltage dip	S	ON
ANLOG	17-Oct	19:00:00	Research Sub	General Alarms	Voltage dip	Z	ON
ANLOG	17-Oct	20:00:00	Research Sub	General Alarms	Voltage dip	X	ON
ANLOG	17-Oct	21:00:00	Research Sub	General Alarms	Voltage dip	X	ON
ANLOG	17-Oct	22:00:00	Research Sub	General Alarms	Voltage dip	S	ON
ANLOG	17-Oct	23:00:00	Research Sub	General Alarms	Voltage dip	S	ON
ANLOG	17-Oct	00:00:00	Research Sub	General Alarms	Voltage dip	Z	ON
ANLOG	17-Oct	01:00:00	Research Sub	General Alarms	Voltage dip	X	ON
ANLOG	17-Oct	02:00:00	Research Sub	General Alarms	Voltage dip	S	ON
ANLOG	17-Oct	03:00:00	Research Sub	General Alarms	Voltage dip	S	ON
ANLOG	17-Oct	04:00:00	Research Sub	General Alarms	Voltage dip	Z	ON
ANLOG	17-Oct	05:01:00	Research Sub	General Alarms	Voltage dip	T	ON
ANLOG	18-Oct	06:01:00	Research Sub	General Alarms	Voltage dip	Z	ON
ANLOG	18-Oct	07:01:00	Research Sub	General Alarms	Voltage dip	S	ON
ANLOG	18-Oct	08:01:00	Research Sub	General Alarms	Voltage dip	Z	ON
ANLOG	18-Oct	09:01:00	Research Sub	General Alarms	Voltage dip	T	ON
ANLOG	18-Oct	10:01:00	Research Sub	General Alarms	Voltage dip	Z	ON
ANLOG	18-Oct	11:01:00	Research Sub	General Alarms	Voltage dip	X	ON
ANLOG	18-Oct	12:01:00	Research Sub	General Alarms	Voltage dip	Z	ON
ANLOG	18-Oct	13:01:00	Research Sub	General Alarms	Voltage dip	Z	ON
ANLOG	18-Oct	14:01:00	Research Sub	General Alarms	Voltage dip	Z	ON

Fig. 6.34. ENMAC alarm view of the generated voltage dips.

The results obtained during these tests show that voltage dips and the alarm displays on ENMAC is now expressed in time frames, as specified by the NRS [28, pp. 8, 9]. The different classes of voltage dips are shown on both the OW Substation diagram page, as well as the alarm page, where:

- ☞ Voltage dip duration of 20-150ms is shown as a type "**X**" dip on ENMAC.
- ☞ Voltage dip duration of 150-600ms is shown as a type "**S**" dip on ENMAC.
- ☞ Voltage dip duration of 600-3000ms is shown as a type "**Z**" dip on ENMAC.
- ☞ A <10% dip is not regarded as a dip.
- ☞ A 10-20% dip is classified as a type "Y" dip.
- ☞ A 20-60% dip is classified as an "X" or "Y" dip, depending on the dip duration.
- ☞ A >60% dip is classified as a "T" or "Z" dip, depending on the dip duration.

Voltage dips can now be represented on the ENMAC, strictly according to the criteria laid down by the NRS [Fig. 3.2, p.38].

The next figure (Fig. 6.35) shows the final layout of the substation on the ENMAC. This shows data captured after the IED was installed, combining the substation plant with the normal substation alarms, indications, analogues and Controls with the outputs from the IED. On this screen the reader can view the IED fundamental, 2^{nd} to 13^{th} Harmonic, THD, RMS and Peak voltage as configured for the IED.

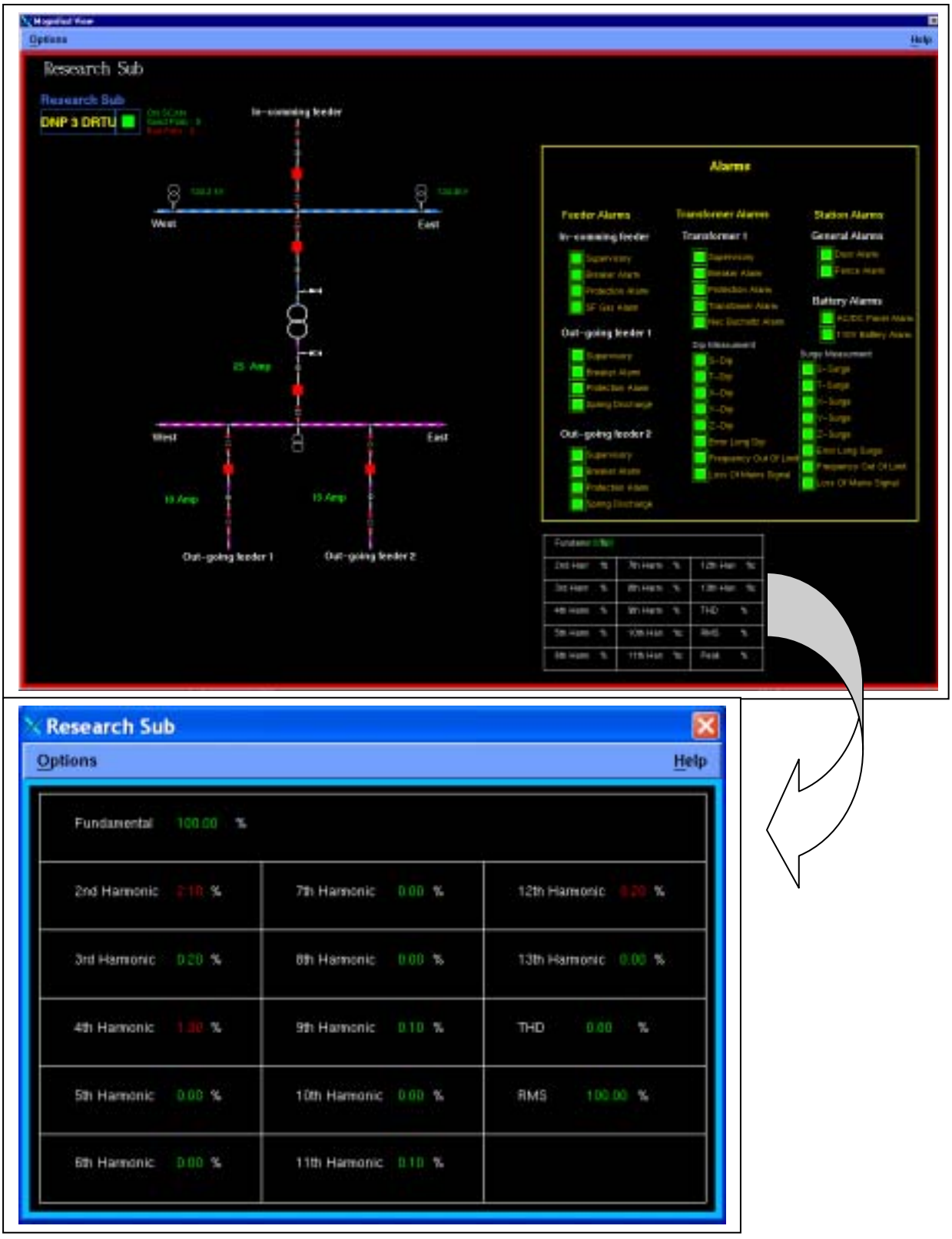


Fig. 6.35. The final layout of the substation on the ENMAC, showing harmonic and voltage dip states.

6.12. Observations during the test period

During all the test phases of the IED prototype, as well as the final development, the author observed that the IED circuit board development was able to:

- Present all the aspects of the quality of electricity supply monitoring specifications, as outlined in the hardware development of both the bread boarded version, as well as the final development of this research according to the NRS specifications.
- These results were confirmed by testing, and have been proven to be within acceptable limits of the NRS. In fact, the measured performance exceeded the original specifications on the harmonic measurements which outlined the requirement of 5%.
- The output of the data to a DRTU QUICC-CARD, using MODBUS protocol was also tested and the results meet the expected criteria.

This completes the research part on the IED prototype development to measure and detect the quality of power supply in the Eskom power networks. On the final development, all three the algorithms developed in this phase:

✧ Harmonic detection.

✧ THD analysis.

✧ Dip/Surge detection.

have been tested in a simulated environment and have been deemed satisfactory for the stipulated requirements.

Also observed during these tests was that the developed IED card interface to the RTU QUICC card, performed within expectation. The VT program connected to the QUICC Controller, as well as the FieldComm program monitoring the messages made it really trouble-free to monitor the diagnostic software that was used to perform the on-line monitoring of data, whilst running diagnostics with the ERTU on-line.

With each variance of the 110VAC supply input, the IED responded well, and all "raw" changes that was visualised on the VT capturing program was forwarded successfully to the ENMAC. Although the transmitting rate of the VT of 9600Bd was sometimes under suspicion due to delays in the VT displays, a change to a slower baud rate of 4800Bd seemed unnecessary, and at the end it was found that all data were correctly processed and it was returned back to 9600Bd.

The FieldComm programme used as a test set for the DNP-03 communications to the ENMAC was of essential help, and without this program it would have been impossible to perform the necessary tests. The configuration on the QUICC for the IED to allow for 16-bit analogue data, with the 13 harmonic inputs, as well as THD, RMS and the error flags, proved to work well. This can be adjusted to the specific need in future, but for the purpose of these tests, it performed acceptably.

CHAPTER 7

SUMMARY AND CONCLUSIONS

7.1. Summary

In this section, a summary of the most important investigations, problems and results encountered during this research are shown.

7.1.1. Power line disturbances

Although electrical utilities try to deliver clean, well-regulated power to their customers, events beyond their control frequently make this impossible. The types of power line disturbances include over-voltages and under-voltages, voltage dips and surges, transients and voltage harmonics (or “noise”) on the power line. These are all voltage phenomena which affect the magnitude and waveform of the AC sine wave. It was found that changes in any load on the system or any primary plant installation might affect interactions with load protection equipment, such as loads on the same feeder that may be added or removed. Providentially these power line disturbances can now be examined and analysed by using the findings and results of this research, incorporating ordinary SCADA techniques.

Prior to this research, power line monitoring could only be carried out on an irregular test basis by using temporarily installed test equipment. With the IED developed during this research, the installation is permanent with the advantage that immediate detection of any

variations in power line characteristics can be achieved. By implementing the IED, and configuring the programmes developed during this research, it is now possible to enable a SCADA system to monitor the load supply characteristics, striving to improve the life span of various high-voltage apparatus, and also those of customer equipment.

It was imperative that harmonics and voltage dips were addressed in line with the National Rationalised Standards (NRS) specifications. Therefore a primary objective of this research was to manipulate the data that are transmitted to the Central Master station into such a format that it can be viewed on the SCADA system in the same format as specified by the NRS. This information is available for access by anyone allowed on to the Local Area Network, and the data shows the quality of supply conditions, which can now be transmitted to relevant persons via Electronic mail and/or Cellular SMS messages for rapid preventative action.

Any harmonic problem used to be very difficult to solve in the region under consideration. That was because harmonic measurements required sophisticated measuring apparatus that is cumbersome and time consuming to be connected to the power system by skilled personnel. Such apparatus was connected directly into the CT or VT circuits, and because of the danger and expertise attended with these connections, authorised personnel were needed to connect apparatus to the power system. With the IED implementation, this risk is eliminated completely.

7.1.2. Hardware assembling

The signal filtering of the mains input was accomplished by using the LTC1063 in a low-pass Butterworth filter implementation. It is clock tuneable, has 12-bit accuracy, low noise, and low dc offset. It was also selected because it provides minimal attenuation of the 13th harmonic at 650Hz, and maximum attenuation of frequencies that will be aliased back onto the used frequencies; e.g. 1% or -40dB.

Considering the analogue to digital conversion, a few aspects were investigated like ADC resolution, accuracy (quantisation error and non-linearity), sampling rate and aliasing. Also a detailed understanding of how an ADC is specified was compulsory before making a final decision. Once the correct architecture, sampling rate and resolution have been finalised the board was laid out properly in order to take full advantage of the performance of the ADC. On choosing the right ADC to fit this particular application, many factors were considered, including:

- How often the input must be sampled.
- What is the accuracy requirement?
- Power dissipation as well as the size of the package.

Selecting the proper ADC for a particular application was not an easy task. Most ADC applications can be classified into market segments like data acquisition, precision industrial measurement and high speed, and a complex understanding of the three most popular ADC architectures and their relationship to the market segments was necessary for the selection.

The AD7864 four-channel sampling 12-bit A/D converter was chosen due to its cost, high speed, low power, and four-channel simultaneous sampling 12-bit A/D converter features.

The MC68HC812A4 microcontroller unit (MCU) was chosen mainly because of the cost and high speed. It is a 16-bit device composed of standard on-chip peripheral modules connected by an inter-module bus. Modules include a 16-bit central processing unit, (CPU12), a Lite integration module (LIM), two asynchronous serial communications interfaces (SCI0 and SCI1), a serial peripheral interface (SPI), a timer and pulse accumulation module, an 8-bit A to D converter (ADC), 1-Kbyte RAM, 4-Kbyte EEPROM, and memory expansion logic with chip selects, key wakeup ports, and a phase-locked loop (PLL).

For evaluation purposes, the Axiom CME12-A4 single board computer was used as it is a fully assembled, fully functional development system for the Motorola 68HC12A4 microcontroller, complete with wall plug style power supply and serial cable, software and documentation.

7.1.3. Software development

The software development on the IED was written predominantly using a C compiler of LINUX. The reason behind this decision was that LINUX have the built-in ability to allow the combination of input and output programs which could help us in testing the results, and it also has the additional benefit of low cost.

The harmonic software was required to detect harmonic components on a 50Hz sinusoidal input wave up to the 13th Harmonic. The routine developed for harmonic detection looked at a frame of samples and calculates the amplitude content, which it then reports. To cut processing time to a minimum, the dip-detection routine used the value that was already calculated for the fundamental frequency. The programs for harmonics and voltage dip/surge calculation as well as the data can be summarised as follows:

- A communications task, which looks for any MODBUS slave address and receives the rest of the messages.
- A calculation task for dips and surges.
- Harmonics calculation task for harmonic measurements.
- Calculation task for the THD.
- RMS calculation task.

7.1.4. IED system performance

During the test phases of the IED prototype, as well as tests on the final development, the author observed that the IED circuit was able to present all the aspects of the Quality-of-Supply monitoring specifications as specified by the NRS. This is true of the hardware development of both the bread boarded version, and the final development. These results were confirmed by testing, and proved to meet the specifications on the harmonic measurements as outlined by the NRS requirements.

On the final development, all three the algorithms developed, harmonic detection, THD Analysis and Dip/Surge Detection, have been tested in a simulated environment and have been deemed agreeable for the specific requirements.

7.1.5. Protocols

Three main protocols that were used during this research:

- Eskom Telecontrol (ESTEL) protocol. This was used to communicate to external devices as the M/IO to the QUICC. It can also be used to communicate between the RTU and ENMAC.
- Distribution Network Protocol version 3 (DNP-03). This protocol was pioneered within ESKOM South Africa during this research, and was used to communicate via the RTU to ENMAC. DNP-03 is a relatively high-overhead protocol, compared to its predecessors, and provides several different means of retrieving data, like Solicited or Unsolicited modes.
- MODBUS protocol. This protocol was also pioneered within ESKOM North Western Region during this research, and was used to interface the IED development to the ERTU Quad Integrated Communication Controller (QUICC) card.

7.2. Significant outcomes from this research project

At the time of writing of this thesis, the IED hardware as developed has sufficient resources in processing power and memory to enable future software modifications for more advanced

control techniques and experimentation. A highly reliable quality of electricity supply measuring system is now operating via the SCADA system, with the distinctiveness of using the relevant substation information received from the IED to be processed via SCADA, and can be categorised as follows:

- Monitoring and localising: Power quality can now be monitored, logged and localised via a standard SCADA system, which was not possible on most major advanced systems world-wide, before this research project was initiated.
- Quality of supply predictions: Historic trends and data on the SCADA system can now be used to make possible distortion forecasts when additional load is applied. This power distortion prediction can be performed before major new electronic systems are installed, using previous data. This is undoubtedly one of the greatest advantages of this project.
- Dip categorisation: It is now possible to categorise dip-levels and pollution, which can assist with the calculation of circuit-breaker settings on specific power lines.

7.3. Assimilation of components and systems

The reader may have observed that this research project not only involved the development of a measuring unit, in this case the IED, but it also required the interfacing to contemporary SCADA units as well as pioneering protocol commissioning. To comply with these criteria, a wide knowledge of existing systems was necessary. To aggravate the situation it was found that a lot of the crucial issues targeted during this research were not introduced on the working SCADA system up till now, and this had to be dealt with.

Firstly, the IED had to be able to interface to existing substation plant equipment. To avoid initial installation risks, care had to be taken on the positioning and interfacing methods of the unit. Secondly, the IED interface to the RTU was taken care of, using RS232 via MODBUS protocol to interface to the RTU QUICC card. Completing the configuration, the RTU had to be configured and commissioned to communicate to the ENMAC via DNP-03 unsolicited. The final phase entailed the ENMAC configuration to enable the IED view of harmonics and voltage dip alarms according to the NRS specifications.

7.4. Overall system effectiveness and economy

From the initial testing performed on the IED as a system, the new type of QOS measuring system had the capacity to exceed the performance of any of the old traditional systems in installation, speed and reliability. Also, with the major reduction in the complexity of the wiring harness, and with the new compact, efficient electronics, the overall system cost is substantially cheaper than any of the old systems, both to construct and operate. The cost of components, connectors, printed circuit board, and development was less than R12 000; while production cost on further boards should not exceed R3000. The cost of the wiring harness is negligible. The labour of installing the old complex, temporary units has been saved, and the new boards are simple to install, resulting in a substantial saving in installation and travelling time compared to the old system. Another feature of the new system is that it is cheaper to operate than the old systems, as there are no moving parts like cooling fans or other parts to wear, and the electronics are highly efficient.

7.5. Project pitfalls

A major problem with the smooth development of this development was the decision to use components that had just been released at the time of starting with the research. This resulted in difficulty obtaining the components in time. Eventually, after a great deal of development was performed on some components, we experienced problems interfacing with the RS232 port, and the interface component was damaged and could not be replaced.

Another problem was the ASDP 2181 boot loader that was used to quickly download a program in the DSP. Due to an oversight in the preliminary design of the original prototype board, a capacitor that was needed on the flash programming power pin was not installed, resulting in some noise on the power rail near the IC, and consequently corruption of the flash during programming. The quick solution at that time was to make use of a JTAG emulation programming pod for the host PC. This communicated with the DSP core via dedicated hardware on the DSP. The delays caused by this problem set the project back a while, but the project suffered no other major problems.

Further, delays in the implementation of the DNP-03 protocol have resulted in an inability to use the integrated capability of some of the integrated Recloser RTU units e.g. NULEC and Form 5, which can only work on DNP-03. These could not be commissioned due to the older protocols, e.g. ESTEL and INTRAC, on the existing channels, causing communications conflicts. This delayed the interfacing of the Intelligent Electronic Devices (IED's) to the RTU's, as the RTU protocol problem had to be solved first. This seriously influenced the goal

of this research, revising the scope to involve DNP-03 protocol configurations to be pioneered by the author along with the IED development.

7.6. Interfacing with other systems

The RS232 port of the IED ensures flexibility in interfacing to other systems like:

- Other SCADA master stations and RTU's.
- GPRS.
- Cellular modems.
- Broadband Radios.

On most RTU's and SCADA master systems, most utilities often desire to manage Telecontrol units using their existing SCADA system and consoles as the primary interface. Protocol converters are available to support SCADA protocols other than MODBUS and DNP. Since most alternate protocols are based on an RS-232 physical link, protocol converters are available and can be located at the utility's SCADA facility near the RS-232 ports used for RTU communications. The IED development have the possibility to interface to most major types of RTU's that have RS232/485 interface ports.

GPRS (General Packet Radio Service) is a step between GSM and 3G cellular networks, and offers faster data transmission via a GSM network within a range of 9.6Kbits to 115Kbits. This technology was already tested on the ENMAC, with a few pilot stations running successfully at the time of this writing. The main benefits of GPRS are that it reserves radio resources only

when there is data to be sent and it reduces reliance on traditional circuit-switched network elements.

Interfacing the IED directly to a GPRS modem, can save the cost of installing an RTU, especially at sites like the end of a power line, or small transformer or sectionalizing points without infrastructure. The IED may then directly communicate to the Master station via MODBUS.

7.7. Recommendations and pointers to further work

Regional statistics showed that the most frequently encountered harmonics in three-phase distribution networks were the odd orders, and that above order 50, harmonics are negligible and measurements are no longer meaningful. It also shows that sufficiently accurate measurements were obtained by measuring harmonics up to order 30. In general, utilities monitor harmonic orders 3, 5, 7, 11 and 13. Generally speaking, harmonic conditioning of the lowest orders (up to 13) is sufficient, but as more comprehensive conditioning takes into account harmonic orders up to 25, it is recommended that this development be adjusted to measure harmonics up to the 25th.

Phase shift measurements were not examined during this research period. The author would be keen on looking into this in the near future.

7.8. The way forward

The production of the IED quality of electricity supply devices will be started before long. Project proposals and project technical evaluations are already on the way. The aim is that this development will be implemented on all Distribution stations in the region within the next two years, followed by implementing this on major Reticulation substations. It will also be considered to be used at problematic rural lines, where no measuring equipment exists.

Immediate research on implementing the unit as a stand alone concept without SCADA facilities will also be investigated.

7.9. Conclusion

Whether it is the effective controlling of power in a facility or the effective measurement of power quality, the results of this research can provide a solution to the problem. In spite of all the problems encountered, the development and integration of the measuring device was a success. It was thoroughly tested in the laboratory, as well as at the Glen Rural substation, and the application will be implemented on various RTU's in the near future.

This project was entirely successful in its aim of producing the hardware for a modern distributed measuring system for the quality of electricity supply incorporating SCADA. More work can be performed on the software before the IED is complete for operation in an industrial environment.

Although it was expected that the IED would be running by the end of 2004, a proper analysis of the system was needed to be done to ensure optimal performance before starting with the production line and the optimal layout of the hardware, to prevent further modifications in future. Implementing the results of this research proved that the evaluation of the fault performance of transmission and distribution lines in terms of quality of supply could be monitored by SCADA. This information is valuable to any electricity supply utility. Eskom is now in a privileged position because this information can be used to predict future voltage dip, surge and harmonic performances at any electricity facility equipped with SCADA, looking at historic trends.

Implementing the results of this research proved that the evaluation of the fault performance of transmission and distribution lines in terms of quality of supply can be monitored by SCADA. This information can be used to predict voltage dip and harmonic performances at certain types of customer facilities, e.g. mines, farms, factories etc.

It is not only possible now to measure distortion and pollution levels on the power network via most SCADA systems, but it is also possible to locate and identify the sources of this contamination with the aid of the SCADA system. The main advantage is that at existing SCADA remote facilities, existing equipment can be used to assist in eliminating sources of power pollution on the electrical power network, without the supplementary cost of other communications media e.g. cellular or satellite for remote sites.

The availability of historical information of contaminated power supply areas and equipment can now be kept on the Eskom SCADA database for future reference and planning purposes.

This is quite a useful tool to investigate possible customer complaints, who typically expect availability of electricity to be 100%. This kind of request is complicated to meet in practice.

Implementing the results of this research enables the electricity distribution facility to:

- Faster response to quality of electricity supply changes - With increased visibility into the electricity supply chain networks, the utility can be more responsive, as it can sense and respond quickly to changes and quickly react on new event occurrences.
- Increased customer satisfaction - By offering a common information framework that supports communication and collaboration, the IED enables the utility to better adapt to and meet customer demands.
- Compliance with regulatory requirements – The utility can now track and monitor the quality of electricity compliance in all areas specified by the NRS.
- Improved cash flow - Information transparency and real-time business intelligence can lead to better control of overall network cost expenditure.
- Higher margins - With the IED, lower operational expenses can lead to improvements in performance and quality.

The implementation of the results of this research can lead to the benefits of a power quality monitoring service e.g.:

- Assist in preventative and predictive maintenance:
- Identify source and frequency of events.
- Establish precise location and timing of events.

- Develop maintenance schedules based on power quality trends.
- Determine the need for mitigation equipment.
- Monitor and trend conditions.
- Examine harmonics, THD, RMS, voltage dip and surges.
- Make decisions based on documented trends.
- Assess sensitivity of process equipment to disturbances.
- Evaluate performance against specifications.
- Benchmark overall system performance.
- Make multi-site comparisons.
- Assist to improve energy rates.

By implementing the results and product of this development, a lot of supplementary and unwanted costs will be eliminated, and the result will, without doubt, benefit any electricity supplier.

APPENDIX 1

1.1. The Goertzel coefficient calculations

Where the Goertzel coefficients must be calculated, we used the following syntax to start COEFGEN.C

c:> coefgen

N > 463

f_sample > 48600

The resulting output show each tone to detect with its associated k(flt), k(int), and k(err) values. The k(flt) value is the floating point value of N*(ftone/fsample); k(int) is the closest integer to k(flt). This integer is the index of the frequency bin for the closest match. If k(flt) and k(int) are equal, they are perfectly matched and there is no leakage loss occurs. Discrepancies between k(flt) and k(int) lead to leakage losses, and this difference is measured in the k(err) variable. The "goodness of fit" of the N value is judged by the largest squared k(err) value of all the individual tone values.

1.2. Digital tone detection

N=463.000000
fs=48600.000000
f_tone[0]= 11025.00 Hz
k(flt)=105.032410
k(int)=105
k(err)= +0.032410
coef(flt)= +0.290734 coef(2.14 hex)=0x129B
f_tone[1]= 12600.00 Hz
k(flt)=120.037041
k(int)=120

```
k(err)= +0.037041
coef(flt)= -0.115286 coef(2.14 hex)=0xF89F
f_tone[2]= 14175.00 Hz
k(flt)=135.041672
k(int)=135
k(err)= +0.041672
coef(flt)= -0.516546 coef(2.14 hex)=0xDEF1
f_tone[3]= 15750.00 Hz
k(flt)=150.046295
k(int)=150
k(err)= +0.046295
coef(flt)= -0.896475 coef(2.14 hex)=0xC6A0
f_tone[4]= 17325.00 Hz
k(flt)=165.050919
k(int)=165
k(err)= +0.050919
coef(flt)= -1.239387 coef(2.14 hex)=0xB0AE
f_tone[5]= 18900.00 Hz
k(flt)=180.055557
k(int)=180
k(err)= +0.055557
coef(flt)= -1.531119 coef(2.14 hex)=0x9E02
f_tone[6]= 20475.00 Hz
k(flt)=195.060181
k(int)=195
k(err)= +0.060181
coef(flt)= -1.759627 coef(2.14 hex)=0x8F62
f_tone[7]= 23175.00 Hz
k(flt)=220.782410
k(int)=221
k(err)= +0.217590
coef(flt)= -1.979731 coef(2.14 hex)=0x814C
```

1.3. The Filter program

Program of filter file to modify a set of samples

```
prog
::::::::::::::
:
integer expectedexit
integer ourabort
integer rc

# shell script to test combinations of dips
runtest() {
expectedexit=$1
ourabort=$2
./wave|./filter f|./anal 2>&1 > /dev/null
```

```
rc=$?
if test $rc -ne $expectedexit
then echo aborting at position $ourabort with rc of $rc
exit 0
fi

}

# test no-dip area
echo "100 32
91 100
" > f
runtest 0 1

echo "100 32
0 30
" > f
runtest 0 2

echo "100 32
0 31
" > f
runtest 0 3

# because we are out-of-phase, we pick this up as 2-cycles at 50% dip
# making this an X.  Not ideal, but acceptable
echo "100 48
0 31
" > f
runtest 2 4

#test Y area
# do not go above 87% because of hysteresis
#long dip test
echo "100 32
87 4000
" > f
runtest 1 5

echo "100 32
87 64
" > f
runtest 1 6

# an out-of-phase dip here close to the limit is seen as just 1 dip and is
# therefore ignored - acceptable
echo "100 48
87 64
" > f
runtest 0 7
```

```
# but extend the dip by another half-cycle and we pick it up as Y
echo "100 48
87 64
87 16
" > f
runtest 1 8

# test the 4 corners of an X dip
echo "100 32
58 32
58 16
" > f
runtest 2 9

# this averages out over 2 cycyles to over 80%, giving a Y - acceptable
echo "100 32
78 32
78 16
" > f
runtest 1 10

# but increase from 1.5 to 2 cycles and we detect properly
echo "100 32
78 32
78 32
" > f
runtest 2 11


echo "100 32
58 32
58 32
58 32
58 32
58 32
58 32
58 32
" > f
runtest 2 12

echo "100 32
78 32
78 32
78 32
78 32
78 32
78 32
78 32
" > f
```

```
runtest 2 13

# now we experimented with the phase
# The result was nasty.  If one have a 1.5 cycle dip just in the X
# region, out of phase by 0.5 cycle, one do not detect it at all.  The energy
# drop in the first half-cycle is not high enough to register as a dip, and
# the second cycle is ignored as it is a 1-dip glitch!
# But this were still acceptable in terms of the design - as we were within the
# 1-cycle deviation limit
echo "100 48
78 32
78 16
" > f
runtest 0 14

# increasing the pulse width by another half-cycle makes us a Y
echo "100 48
78 32
78 32
" > f
runtest 1 15

# and another half-cycle makes us see this as an X
echo "100 48
78 32
78 48
" > f
runtest 2 16

# detect the S dip
# test the 4 corners of an X dip
# 8 times 32 = 256
echo "100 32
58 256
" > f
runtest 3 17

echo "100 32
78 256
" > f
runtest 3 18

echo "100 32
58 950
" > f
runtest 3 19

echo "100 32
78 950
" > f
```

```
runtest 3 20

# alter the phase a bit, but after 8 cycles 1 does not make much difference
echo "100 48
58 950
" > f
runtest 3 21

echo "100 48
78 950
" > f
runtest 3 22

echo "100 60
78 950
" > f
runtest 3 23

# T test
# check the corners
# notice that we need 2 full cycles at this energy - any less and we see it as
# an X
echo "100 32
38 64
" > f
runtest 4 24

# and many cycles for worst-case phase if we are at the edge of the voltage
echo "100 48
37 256
37 256
37 256
" > f
runtest 4 25

# detection is fine at 30 cycles
echo "100 32
38 958
" > f
runtest 4 26

# even with a phase shift
# although here, we slide into a Z because the phase shift adds a sample
echo "100 48
38 958
" > f
runtest 5 27

echo "100 32
05 958
```

```
" > f
runtest 4 28

echo "100 48
00 940
" > f
runtest 4 29

echo "100 48
00 48
" > f
runtest 4 30

# Z test
# check the corners
echo "100 32
00 970
" > f
runtest 5 31

echo "100 48
00 970
" > f
runtest 5 32

echo "100 32
00 3000
" > f
runtest 5 33

echo "100 48
00 3000
" > f
runtest 5 34
```

1.4. The Make files

Type “make” to build the entire system.

```
::::::::::::::
hdr.h
::::::::::::::
**********
General definitions
*********/

#define NUM_HARMONICS 16        /* ideally, a power of 2 */
#define SAMPLE_RATE (2*NUM_HARMONICS)
```

```
#define FUNDAMENTALFREQ 50

#ifndef PI
#define PI 3.14159265358979323846264 34
#endif

/* function prototypes */
int output (unsigned x, double y);
unsigned get_timesample(unsigned fund, unsigned harmonics);
short getsample();
outputsample(short s);
diplog(double fund_ampl);
int calc_dip_state();
double calc_thd(double *, int);
```

1.5. The Wave files

```
::::::::::::::
wave.h
::::::::::::::
/*************************************************************************
**********
Wave simulation classes.
The sinewave class encapsulates a sinewave and allows the following:
a) Setting of amplitude (in constructor) - recommend 100 for fundamental
b) Increment time (microseconds) in the time domain
c) Get amplitude for the sinewave as it stands in the time domain.
The wave class is a collection of sine waves.
Once the waves have been added, it allows similar functionality to the sine
wave - time increment and amplitude extraction.
*************************************************************************
*********/

#include <stdio.h>
#include <limits.h>
#include <math.h>
#include <vector>
using namespace std;

class sinewave
{
private:
        unsigned freq; // our frequency, in Hz
        unsigned period;        // period, in microsecods, derived from freq
        unsigned amplitude;   // amplitude of this harmonic, in %
        unsigned delta;                 // time offset in microseconds
public:
        sinewave(unsigned hz, unsigned ampl=100);
        void steptime(unsigned micros);       // jump to a new sample time-cursor
```

```
        double getsinevalue(int amplitude_adjust=0);
};

class wave
{
private:
        vector <sinewave *> swv;
        vector <sinewave *>::iterator it;
        int numh;       // number of harmonics
public:
        wave::wave(void) {numh=0; }
        void steptime(unsigned micros);
        double getsinevalue(int amplitude_adjust=0);
        void add_harmonic(unsigned hz, unsigned ampl=100);

};

::::::::::::::
anal.c
::::::::::::::
/*****************************************************************************
**********
Wave analysis program
*****************************************************************************
*********/

#include <stdio.h>
#include <limits.h>
#include "hdr.h"

/* global variables */
short xsamples[SAMPLE_RATE];
double harmonic_amplitudes[NUM_HARMONICS];
double THD;

double fund_freq;


/*
store the sample given, and if we have enough to do an fft,
return 1, else return 0
*/

int store_sample(short value)
{
        static int rc=0;
        static int idx=0;       // index into samples array
        static int ignore=0;    // ignore this number of samples at beginning to offset
phase
```

```
        if (ignore)       {
                ignore--;
                return 0;
        }

        xsamples[idx++]=value;
        if (idx >= SAMPLE_RATE)  {
                idx = 0;
                return 1;
        }
        return 0;
}

double detect(int harmonic)
{
        double rc=0.0;
        int i=0;
        static float denominator=1.0;  // simply used to scale printf output to 100%

        for (i=0; i<SAMPLE_RATE; i++)     {
                rc += (double)xsamples[i]/(double)SHRT_MAX
                        * sin (2 * PI * (harmonic+1) * i / SAMPLE_RATE);
        }
        if (harmonic == 0)
                denominator = rc;

        // printf ("harmonic %2d._Value %.6f_%.3f%\n", harmonic, rc,
100.0*rc/denominator);
        if (rc < 0)
                return -rc;
        return rc;
}

/* given the middle (guess) freqency, scan +- 5Hz to get best fit */
/* this function does not work! */
double findfund(double mid)
{
        double d;
        double bestfreq=0.0;
        double bestfreqval=0.0;
        double freqval=0.0;
        int i;
        double period;

        for (d=mid-5.0; d<(mid+5.0); d+=0.5)          {
        freqval=0.0;
        period = 1000000.0/d;

printf ("%.1f Hz\n", d);
        for (i=0; i<11;/*SAMPLE_RATE;*/ i++)     {
```

```
                freqval // += (double)xsamples[i] //(double)SHRT_MAX
                = sin (2.0 * PI * (double)i * (double)625 /period);
printf ("\t%02d)\t%04x\t%.4f\t%.4f\n",i, xsamples[i], freqval,
xsamples[i]/freqval);
        }
        if (freqval > bestfreqval)      {
                bestfreq=d;
                bestfreqval=freqval;
        }
printf ("freq %.1f Hz, sum=%.2f\n",d,freqval);
        }
printf ("Best Frequency = %.2f\n", bestfreq);
}

main()
{
        unsigned value;
        int i;
        double d;

        while ((value=getsample()) != EOF)  {
                if (!store_sample(value))
                        continue;

                /* calculate the harmonic content of the wave */
                for(i=0;i<NUM_HARMONICS;i++) {
                        d=detect(i);
                        harmonic_amplitudes[i]=d;
                        if (d > 0.01)
                                printf ("harmonic %2d. Value %.6f\n", i, d);
                }

                /* now work out our THD */
                THD=calc_thd(harmonic_amplitudes, NUM_HARMONICS);
                printf ("\tTHD=%.4f\%\n", THD);
                /* process the dip information */
                // printf("Fundamental magnitude: %.2f\n", harmonic_amplitudes[0]);
                i=diplog(harmonic_amplitudes[0]);
                if (i)    {
                        printf ("Dip Type %c\n", i);
#ifdef SHELL_TEST /* shell testing uses exit codes */
                        if (i=='Y') exit(1);
                        if (i=='X') exit(2);
                        if (i=='S') exit(3);
                        if (i=='T') exit(4);
                        if (i=='Z') exit(5);
#endif
                }
        }
}
```

1.6. The dip.c detection module

::::::::::::::
dip.c
::::::::::::::

```
/*****************************************************************************
**********
dip detection module
*****************************************************************************
*********/
#include "hdr.h"

#define IN_DIP          1
#define OUT_DIP         2
#define NUM_DIPS 151

/* global variables */
double fund_peak;       /* peak value for fundamental */
double dip_buf[NUM_DIPS];           /* units are wave-periods - 20 ms each */
int dip_idx;                /* index into buffer */
int state;                  /* our present state - IN_DIP or OUT_DIP */

#define SAVE_DIP(ampl)      { dip_buf[dip_idx++]=100*ampl/fund_peak; \
                        if (dip_idx==NUM_DIPS-1) dip_idx--; } /* no overflow */

/* returns dip-type, or 0 for no dip */
/* but only say we are in a dip if we have just come out of one! */
diplog(double fund_ampl)
{
        int rc;

        if (fund_peak == 0)     {
                fund_peak=fund_ampl;
                state=OUT_DIP;
                return 0;
        }

        /* scale our peak amplitude over a few cycles.... */
        if (fund_ampl > 0.9 * fund_peak)        {
                /* adjust fund_peak by 20% upwards, 10% downwards */
                if (fund_ampl > fund_peak)
                        fund_peak = 0.8 * fund_peak + 0.2 * fund_ampl;
                else
                        fund_peak = 0.9 * fund_peak + 0.1 * fund_ampl;
        }

        if (state == OUT_DIP)           {
                if (fund_ampl < 0.88 * fund_peak)     {
                        state=IN_DIP;
                        SAVE_DIP(fund_ampl);
```

```
                }
                        return 0;
        }
        if (state == IN_DIP)    {
                if (fund_ampl > 0.92 * fund_peak)     {
                        rc = calc_dip_state();
                        state=OUT_DIP;
                        dip_idx=0;
                        return rc;
                }
                SAVE_DIP(fund_ampl);
                state=IN_DIP;
                return 0;
        }
}


int calc_dip_state()
{
        double dip_pc=0.0;     /* average percentage of signal during the dip */
        int i;

        for (i=0; i<dip_idx; i++)
                dip_pc += dip_buf[i];
        dip_pc /= dip_idx;

/*
        printf ("Max=%.2f ", fund_peak);
        for (i=0; i<dip_idx; i++)
                printf("peak%d=%.2f ", i, dip_buf[i]);
        printf("\n");
        printf("(%d dips, %.2f\%)\n", dip_idx, dip_pc);
*/

        /* filter off the noise */
        if (dip_pc >= 90)       /* average dip less than 10% - ignore */
                return 0;
        if (dip_idx < 2)        /* only one cycle - ignore */
                return 0;
#ifdef NOGOOD
        if (dip_idx == 2 && dip_pc >= 50) /* dip overlaps period boundary */
                return 0;
#endif

        /* do we have a type 'Y' ? */
        if (dip_pc >= 80)
                return 'Y';

        /* do we have a type 'Z' ? */
        if (dip_idx > 30)       /* 600 ms plus */
```

```
                return 'Z';

        /* do we have a type 'T' ? */
        if (dip_pc < 40)
                return 'T';

        /* do we have a type 'S' ? */
        if (dip_idx >= 8)       /* 160 ms plus */
                return 'S';

        /* we must have a type 'X' ? */
                return 'X';
}
```

1.7. The filter.c program to simulate dips

```
::::::::::::::
filter.c
::::::::::::::
/*****************************************************************************
**********
filter program to re-shape 16-bit input samples and thereby simulate dips
*****************************************************************************
*********/
#include <stdio.h>
#include "hdr.h"

#define ARRSIZE 10000
short filterarr[ARRSIZE];      /* array of modifier numbers - 100 = 100 % */
int filterarrsize;

addfilter (int value, int occurrences)
{
        if ((filterarrsize + occurrences) >= ARRSIZE)
                occurrences = ARRSIZE - filterarrsize - 1;

        if (occurrences <= 0)
                return 0;

        while(occurrences--)  {
                filterarr[filterarrsize++]=value;
        }
}

fillfilter()       /* fill the rest of the filter array with 100's */
{
        while(filterarrsize < ARRSIZE)
                filterarr[filterarrsize++]=100;
}
```

```
short modifysample(short s)
{
	double d;	/* intermediate scaled value */
	d = (double)s * (double)filterarr[filterarrsize++]/100.0;
	if (filterarrsize == ARRSIZE){
		filterarrsize--;
		return s;
	}
	return (short) d;
}

main(int argc, char *argv[])
{
	char linebuf[80];
	FILE *fp;
	int i;
	int j=1;
	short sample;

	if (argc != 2)	{
		fprintf (stderr, "usage: %s filter_file_name\n", argv[0]);
		exit(1);
	}

	if ((fp=fopen(argv[1], "r"))==NULL){
		fprintf(stderr, "could not open file %s\n", argv[1]);
		exit(1);
	}

	while (fgets(linebuf, 80, fp))	{
		if (linebuf[0]=='#')
			continue;	/* we have a comment */
		if (linebuf[0] < '0'  || linebuf[0] > '9')
			continue;	/* unexpected input */
		i=j=1;  /* reset them in case they are not specified */
		sscanf (linebuf, "%d%d", &i, &j);

		/* printf ("i=%d, j=%d\n", i, j); */
		addfilter (i, j);
	}

	fillfilter();

	/* testing loop - works okay
	for (i=0; i<filterarrsize; i++)
		printf ("%d ",filterarr[i]);
	*/

	filterarrsize=0;/* reset index */
	while ((sample=getsample()) != EOF)	{
```

```
                sample=modifysample(sample);
                outputsample(sample);
        }
}
```

1.8. Implementation of the wave classes

```
::::::::::::::
wave.cpp
::::::::::::::
/*****************************************************************************
**********
Implementation of the wave classes in wave.h
*****************************************************************************
*********/
#include "wave.h"
#include "hdr.h"

/* increment all harmonics by the given value */
void wave::steptime(unsigned micros)
{
        for (it=swv.begin(); it != swv.end(); ++it)
                (*it) -> steptime(micros);
}

double wave::getsinevalue(int amplitude_adjust)
{
        double rc=0;

        for (it=swv.begin(); it != swv.end(); ++it)
                rc += (*it) -> getsinevalue(amplitude_adjust);
        // cout << "Sine value = " << rc << endl;

        return rc;
}

void wave::add_harmonic(unsigned hz, unsigned ampl)
{
        // cout << hz <<" Hz, ampl=" << ampl << endl;
        swv.push_back(new sinewave(hz, ampl));
        // cout << "size=" << swv.size() << endl;

}


sinewave::sinewave(unsigned hz, unsigned ampl)
{
        freq=hz;
        amplitude=ampl;
        delta=0;
```

```
        if (freq==0)    {
                cerr << "Frequency of 0 is illegal\n";
        }

        period = 1000000/freq;
        // period = 1000000/50; // fixed sample rate
}

void sinewave::steptime(unsigned micros)
{
        delta=(delta+micros)%period;
        return;
}

double sinewave::getsinevalue(int amplitude_adjust)
{
        double sineval, rads;

        rads = (2.0 * PI) * (double)delta/(double)period;
        sineval = sin(rads);
        if (amplitude_adjust)
                sineval *= ((double)amplitude/(double)100);
        return sineval;
}

::::::::::::::
wavesim.cpp
::::::::::::::
/*************************************************************************
**********
Simulation program to generate wave train, with harmonics, and produce
16-bit hex output.
*************************************************************************
*********/

#include "wave.h"
#include "hdr.h"

/* global variable definitions */
unsigned harmonic_amplitude[NUM_HARMONICS];

main()
{
        /* set the amplitude of the harmonics that we want */
        /* set one frequency to 100 (fundamental, usually) and
        then the others to a percentage of that.  A scaling
        factor to avoid overflow will be calculated later */

        harmonic_amplitude[0]=100;
```

```
        harmonic_amplitude[4]=8;
        harmonic_amplitude[8]=22;
        harmonic_amplitude[12]=4;
/*
        harmonic_amplitude[0]=100;
        harmonic_amplitude[1]=20;
        harmonic_amplitude[2]=15;
        harmonic_amplitude[3]=12;
        harmonic_amplitude[4]=10;
        harmonic_amplitude[5]=8;
        harmonic_amplitude[6]=7;
        harmonic_amplitude[7]=6;
        harmonic_amplitude[8]=5;
*/
        unsigned u;
        unsigned timesample;
        int i;
        wave wv;
        double d;

        /* get time sample increment for x-axis */
        timesample=get_timesample(FUNDAMENTALFREQ,
NUM_HARMONICS);
        timesample=625;       // hard-code this preiod for the moment

        /* build up our wave (wv) with its harmonics */
        for (u=0; u<NUM_HARMONICS; u++)
                wv.add_harmonic((unsigned)(FUNDAMENTALFREQ*(u+1)),
                        harmonic_amplitude[u]);

        /* output a continual stream of samples */
        for (u=0; u<6000; u++)        {
                d=wv.getsinevalue(1);          // amplitude-adjusted value
                output (u*timesample, d);
                wv.steptime(timesample);      // increment time sample
        }
}

/* for a given x time-value, this is the wave amplitude */
output (unsigned x, double y)
{
        int i;
        double scalefactor=0.0;
        short outvalue;// 16-bit output value

        /* the amplitude can exceed 1 because of the summation
        of the harmonic amplitudes.  If this happens, we overflow
        the output.  So we calculate a crude scaling factor, by adding
        the amplitdes,
        */
```

```
        for (i=0; i<NUM_HARMONICS; i++)
                scalefactor += (double)harmonic_amplitude[i];
        scalefactor /= 100.0;

        y /= scalefactor;          // scale our y-value
        y *= (double)SHRT_MAX;

        outvalue=(short)y;
        printf ("%04x\n", outvalue&0xFFFF); // 16-bit output
        // printf ("%2u millis: ampl=%.4f\n", x, y);
}
```

1.9. The “make” file

```
::::::::::::::
Makefile
::::::::::::::
all:    wave anal filter
wave:  wavesim.o wave.o lib.o wave.h hdr.h
        g++ wave.o wavesim.o lib.o -lm -o $@
filter:  filter.o lib.o hdr.h
        g++ filter.o lib.o -lm -o $@
anal:   anal.o dip.o lib.o hdr.h
        g++ anal.o dip.o lib.o -lm -o $@
wavesim.o:     wavesim.cpp wave.h hdr.h
        g++ -c wavesim.cpp
wave.o:         wave.cpp wave.h hdr.h
        g++ -c wave.cpp
lib.o:  lib.c hdr.h
        g++ -c lib.c
anal.o: anal.c hdr.h
        g++ -c anal.c
filter.o:filter.c hdr.h
        g++ -c filter.c
dip.o:  dip.c hdr.h
        g++ -c dip.c

::::::::::::::
prog
::::::::::::::
:
integer expectedexit
integer ourabort
integer rc

# shell script to test combinations of dips
runtest() {
expectedexit=$1
ourabort=$2
```

```
./wave|./filter f|./anal 2>&1 > /dev/null
rc=$?
if test $rc -ne $expectedexit
then echo aborting at position $ourabort with rc of $rc
exit 0
fi

}

# test no-dip area
echo "100 32
91 100
" > f
runtest 0 1

echo "100 32
0 30
" > f
runtest 0 2

echo "100 32
0 31
" > f
runtest 0 3

# because we are out-of-phase, we pick this up as 2-cycles at 50% dip
# making this an X.  Not ideal, but acceptable
echo "100 48
0 31
" > f
runtest 2 4

#test Y area
# do not go above 87% because of hysteresis
#long dip test
echo "100 32
87 4000
" > f
runtest 1 5

echo "100 32
87 64
" > f
runtest 1 6

# an out-of-phase dip here close to the limit is seen as just 1 dip and is
# therefore ignored - acceptable
echo "100 48
87 64
" > f
```

```
runtest 0 7

# but extend the dip by another half-cycle and we pick it up as Y
echo "100 48
87 64
87 16
" > f
runtest 1 8

# test the 4 corners of an X dip
echo "100 32
58 32
58 16
" > f
runtest 2 9

# this averages out over 2 cycyles to over 80%, giving a Y - acceptable
echo "100 32
78 32
78 16
" > f
runtest 1 10

# but increase from 1.5 to 2 cycles and we detect properly
echo "100 32
78 32
78 32
" > f
runtest 2 11


echo "100 32
58 32
58 32
58 32
58 32
58 32
58 32
58 32
" > f
runtest 2 12

echo "100 32
78 32
78 32
78 32
78 32
78 32
78 32
78 32
```

```
" > f
runtest 2 13

# now I experimented with the phase
# The result is nasty.  If one have a 1.5 cycle dip just in the X
# region, out of phase by 0.5 cycle, one do not detect it at all.  The energy
# drop in the first half-cycle is not high enough to register as a dip, and
# the second cycle is ignored as it is a 1-dip glitch!
# But this are still acceptable in terms of the design - as I am within the
# 1-cycle deviation limit
echo "100 48
78 32
78 16
" > f
runtest 0 14

# increasing the pulse width by another half-cycle makes us a Y
echo "100 48
78 32
78 32
" > f
runtest 1 15

# and another half-cycle makes us see this as an X
echo "100 48
78 32
78 48
" > f
runtest 2 16

# detect the S dip
# test the 4 corners of an X dip
# 8 times 32 = 256
echo "100 32
58 256
" > f
runtest 3 17

echo "100 32
78 256
" > f
runtest 3 18

echo "100 32
58 950
" > f
runtest 3 19

echo "100 32
78 950
```

```
" > f
runtest 3 20

# alter the phase a bit, but after 8 cycles 1 does not make much difference
echo "100 48
58 950
" > f
runtest 3 21

echo "100 48
78 950
" > f
runtest 3 22

echo "100 60
78 950
" > f
runtest 3 23

# T test
# check the corners
# notice that we need 2 full cycles at this energy - any less and we see it as
# an X
echo "100 32
38 64
" > f
runtest 4 24

# and many cycles for worst-case phase if we are at the edge of the voltage
echo "100 48
37 256
37 256
37 256
" > f
runtest 4 25

# detection is fine at 30 cycles
echo "100 32
38 958
" > f
runtest 4 26

# even with a phase shift
# although here, we slide into a Z because the phase shift adds a sample
echo "100 48
38 958
" > f
runtest 5 27

echo "100 32
```

```
05 958
" > f
runtest 4 28

echo "100 48
00 940
" > f
runtest 4 29

echo "100 48
00 48
" > f
runtest 4 30

# Z test
# check the corners
echo "100 32
00 970
" > f
runtest 5 31

echo "100 48
00 970
" > f
runtest 5 32

echo "100 32
00 3000
" > f
runtest 5 33

echo "100 48
00 3000
" > f
runtest 5 34
```

1.10. Program bestn.c

Program bestn.c

```
#include <string.h>
#include <stdio.h>
#include "tones.def"

int round( x )
float x;
{
   if (x>0) return ( (int) (x+0.5) );
   else if (x<0) return ( (int) (x-0.5) );
   else if (x==0) return ( 0 );
   else printf("\7bad data in round() function!");
```

```
   return(-1);
}

char f1name[]="bestn.err";
char f2name[]="bestn.N";

main(argc,argv)
int argc;
char **argv;
{
   int i, N, minN, maxN, kint;
   float f_sample, kflt, errsqr, maxerrsqr, detect, binwidth;
   FILE *f1, *f2;

   if (argc!=4)
      {
      printf("\nusage: %s <f_sample Hz [%%f] > <minN [%%d] > <maxN [%%d]
>\n",argv[0]);
      return(-1);
      }
   sscanf(argv[1],"%f",&f_sample);
   sscanf(argv[2],"%d",&minN);
   sscanf(argv[3],"%d",&maxN);
   printf("*** scanning N from %d to %d, where f_sample = %f Hz
***\n",minN,maxN,f_sample);
   if (minN>=maxN)
      {
      printf("minN>=maxN!");
      return(-1);
      }

   f1=fopen(f1name,"w"); if (f1==NULL) printf("\nerror opening %s\n",f1);
   f2=fopen(f2name,"w"); if (f2==NULL) printf("\nerror opening %s\n",f2);
   for (N=minN; N<=maxN; N++)
      {
      maxerrsqr=0;
      binwidth=f_sample/(float)N;
      for (i=0; i<HOW_MANY_FREQUENCIES; i++)
         {
         kflt=((float)(N))*(f_freq[i]/f_sample);
         kint=round(kflt);
         errsqr=(kflt-(float)(kint))*(kflt-(float)(kint));
         if (errsqr>maxerrsqr) maxerrsqr=errsqr;
         }
      detect=((float)(N)*1000.0)/f_sample;
      printf("\n%f =maxerrsqr ",maxerrsqr);
      printf("(N=%6d) ",N);
      printf("detect= %10.3f ms ",detect);
      printf("resolu= %10.3f Hz",binwidth);
      if ((f1)&&(f2))
```

```
            {
            fprintf(f1,"%f\n",maxerrsqr);
            fprintf(f2,"%d\n",N);
            }
        }
    printf("\nyou may want to pipe output to sort utility");
    printf("\nsort according to incr maxerrsqr");
    printf("\nor plot %s vs %s\n",f1name,f2name);
    fclose(f1);
    fclose(f2);
}
```

1.11. Program coefgen.c

```
#include <math.h>
#include <stdio.h>
#include "tones.def"

int round( x )
float x;
{
    if (x>0) return ( (int) (x+0.5) );
    else if (x<0) return ( (int) (x-0.5) );
    else if (x==0) return ( 0 );
    else printf("\7bad data in round() function!");
    return(-1);
}

int flt_to_Q15( x, txt )
float x;
char txt[];
{
   int i, err=0;

   i = round(x*32768.0);
   if (x>=1.0)   { i=0x7FFF; err=1; }
   if (x<(-1.0)) { i=0x8000; err=(-1); }
   sprintf( txt, "%08X\n", i );
   txt[0]=txt[4]; txt[1]=txt[5]; txt[2]=txt[6]; txt[3]=txt[7]; txt[4]='\000';
   return(err);
}

main(argc,argv)
int argc;
char **argv;
{
    int i, kint;
    float N, f_sample, kflt, kerr, coef;
    char Q15coef[255];
```

```
   switch(argc) /* get missing arguments */
      {
         case 1:     printf("N > ");        scanf("%f",&N);
      case 2:     printf("f_sample > "); scanf("%f",&f_sample);
      case 3:     break;
      default:    printf("\n\7usage: %s <N [%%f]> <f_sample [%%f]>\n"); return(-1);
      }
   switch(argc) /* read the arguments */
      {
      case 3:     sscanf(argv[2],"%f",&f_sample);
      case 2:     sscanf(argv[1],"%f",&N);
      case 1:     break;
      }
   printf("\nN=%f\nfs=%f\n",N,f_sample);
   for (i=0; i<HOW_MANY_TONES; i++)
      {
      kflt=N*(f_freq[i]/f_sample);
      kint=round(kflt);
      kerr=kflt-(float)(kint);
      coef=2*cos((2*PI*(float)kint)/N);
      flt_to_Q15( coef/2, Q15coef );
      printf("\nf_freq[%2d]=%10.2f Hz",i,f_freq[i]);
      printf("\n\t\tk(flt)=%10.6f",kflt);
      printf("\n\t\tk(int)=%4d",kint);
      printf("\n\t\tk(err)=%+10.6f",kerr);
      printf("\n\t\tcoef(flt)=%+10.6f coef(2.14 hex)=0x%s",coef,Q15coef);
      }
   printf("\n");
}
Program factor.c
#include <stdio.h>

int prime[2000], factor[2000];
FILE *fp;

int isaprime( a, howmany )
int a, howmany;
{
  int i;
  for (i=0; i<howmany; i++)
    if (a==prime[i]) return(1);
  return(0);
}

int findaprime( a, howmany )
int a, howmany;
{
  int i;
  for (i=1; i<howmany; i++)
    if ((a%prime[i])==0) return(prime[i]);
```

```
  return(0);
}

main(argc,argv)
int argc; char **argv;
{
  int i, j, orig, freq, num, maxprimes;

  if (argc!=2)
    {
    printf("number to factor> ");
    scanf("%d",&freq);
    }
  else
    sscanf(argv[1],"%d",&freq);

  fp=fopen("primes.dat","r");
  if (fp==NULL) {printf("\nerror opening primes.dat\n");return(-1);}
  i=0;
  while (!feof(fp)) fscanf(fp,"%d",&prime[i++]);
  maxprimes=i-1;
  printf("\n%d primes read\n", maxprimes);
  fclose(fp);

  orig=freq;
  i=0;
  while(1)
    {
    if (isaprime(freq,maxprimes)==1) { factor[i++]=freq; break; }
    num=findaprime(freq,maxprimes);
    freq=freq/num;
    factor[i++]=num;
    }
  printf("\n %d factored out = ", orig);
  for (j=0; j<i; j++) printf("%d ", factor[j]);
  printf("\n");
}

Program primes.c
#include <stdio.h>

int fact, num, k, prime[1000];
FILE *fp;

main()
{
  fp=fopen("primes.dat","w");
  fprintf(fp,"%4d\n",1);
  fprintf(fp,"%4d\n",2);
  num=2;
```

```
 k=0;
 while(num<=2000)
  {
  fact=num-1;
  while((num/fact*fact)!=num)
   {
   --fact;
   if (fact<=1)
    {
    prime[k] = num;
    fprintf(fp,"%4d\n",prime[k]);
    k++;
    }
   }
  printf("\r%d",num);
  ++num;
  }
 printf("\nDONE.\n");
}
```

1.12. Program bestfs.c

```
#include <string.h>
#include <stdio.h>
#include "tones.def"

main()
{
        int i, j, kint;
        float f_min, f_max, f_incr, f_sample;
        float kflt, maxerrsqr, errsqr;

        for (j=0; possible_sample_freqs[j]>0.1; j++) {
                f_min = possible_sample_freqs[j]-50;
                f_max = possible_sample_freqs[j]+50;
                f_incr = 10;

                printf("\n*** scanning f_sample from %.1f Hz to %.1f Hz, stepping
%.1f Hz ***\n",f_min,f_max,f_incr);
                if (f_min>=f_max)
                {
                        printf("f_min>=f_max!");
                        return(-1);
                }
                else if ((f_max<f_min+f_incr)||(f_incr<=0))
                {
                        printf("bad f_incr value!");
                        return(-1);
                }
                else for (i=0; i<HOW_MANY_FREQUENCIES; i++)
```

```
		{
			if (f_min<(2*f_freq[i]))
			{
				printf("\nNyquist violation: f_freq=%.1f at
f_sampling=%.1f\n",f_freq[i],f_min);
			}
		}
		for (f_sample=f_min; f_sample<=f_max; f_sample=f_sample+f_incr)
		{
			maxerrsqr=0;
			for (i=0; i<HOW_MANY_FREQUENCIES; i++)
			{
				kflt=f_sample/f_freq[i];
				kint=round(kflt);
				errsqr=(kflt-(float)(kint))*(kflt-(float)(kint));
				if (errsqr>maxerrsqr) maxerrsqr=errsqr;
			}
			printf("\n%f = maxerrsqr     (at f_sample = %.1f)", maxerrsqr,
f_sample);
		}
	}
	printf("\nyou may want to pipe output to sort utility");
}

int round( x )
float x;
{
	if (x>0) return ( (int) (x+0.5) );
	else if (x<0) return ( (int) (x-0.5) );
	else if (x==0) return ( 0 );
	else printf("\7bad data in round() function!");
	return(-1);
}
```

REFERENCES

1. Arrillaga, J and Arnold, C.P. Computer Modeling of Electrical Power Systems. John Wiley & Sons, New Zealand, University of Canterbury. 1983.

2. Babst, J. Digital Signal Processing Applications using the ADSP-2100 Family. Applications Staff of Analog Devices, DSP Division, Prentice Hall, Englewood Cliffs. 2001.

3. Boulet, B; Hacksel, P.J; Wikston, J.M. First South African Power Quality Conference. Sheridan Science and Technology Park, Canada, Mississauga. 1999.

4. Brault, J.W. and White, O.R. The analysis and restoration of astronomical data via the Fast Fourier Transform. Astron & Atrophy's, New Delhi. 1971.

5. Bringham, E.O. The Fast Fourier Transform and its Applications. CITY Prentice-Hall, Englewood Cliffs. 1988.

6. Candy, J.V. Signal Processing – The Modern Approach. Electrical Engineering Series, McGraw-Hill International, New York. 1988.

7. Comfort, S. DNP3.0 Training Manual. Issue 1.0. IST Energy, Pretoria. 2001.

8. Dams, A.P; Terblanche, W; Lennox, K.M, Laniewski, G. ESTEL Telecontrol Protocol Specification (master/slave) SCADA application layer definition. Document 2 of 6,Eskom Transmission group, South Africa, Pretoria. 1994.

9. Dorf, R.C. The Electrical Engineering Handbook. CRC Press, CITY University of California. 1993.

10. Dugan, R.C and McGranaghan, M.F. Electric Power System Harmonics Design Guide. McGraw-Edison Power systems. Pennsylvania. 1987.

11. Edison, N.J. 8010 P.Q.Node, PASS User's Guide. BMI, North Carolina, Dranetz. 1993.

12. Elmore W.A. Protective Relaying Theory and Applications. ABB Powertech T&D Company Inc., Coral Springs, Florida, New York. 1994.

13. Enslin, J.H.R. Distortion in power systems. Department of Electrical Engineering, Pretoria, University of Pretoria, 1988.

14. Franklin, A.C and Franklin, D.P. The J&P Transformer Book. A Practical technology of the Power Transformer. 11th ed. Butterworth, London. 1983.

15. Freeman, P.J. Electric Power Transmission and Distribution. 2nd ed. Harrap, Sydney. 1981.

16. Gross, C.A. Power System Analysis. John Wiley & Sons, Auburn University, New York.1979.

17. Guide to the application of Engineering Recommendation. The Assessment of Harmonic Voltage Distortion and Connection of Nonlinear Equipment to the Electricity Supply System in the UK. Engineering Technical Report, New York. 2002.

18. Harris, G.E. ENMAC system administration Training, Module 5. RTAP Issues. Scotland, Edinburgh. 1999.

19. Harris, G.E. ENMAC System Administration Training. RTAP Issues. Ver. 1.0, Scotland, Edinburgh. 1999.

20. Hughes, E. Electrical Technology. 5th Ed. Longman. London and New York. 1981.

21. IEEE Recommended Practice for Industrial and Commercial Power System Analysis. New York, John Wiley & Sons, 1980.

22. Ifeachor, E.C and Jervis, B.W. Digital Signal Processing. A practical Approach. Addison-Wesley Publishers Ltd. Vienna. 1993.

23. Introducing the MODBUS Protocol. The Crompton Instruments Guide to RS485. Communications on the MODBUS Protocol. Crompton Instruments Ltd., Hong Kong. 2002.

24. Isolated Input Analogue to Digital Converter, Technical Description. Issue 1. Telkor, Pretoria. 1986.

25. James, J.G. An Investigation of Harmonic Content on an Electrical Network. University of Queensland, School of Information Technology and Electrical Engineering, Queensland. 2001.

26. Laughton, M.A. and & Say, M.G. Electrical Engineer's Reference Book. Butterworth, London. 1986.

27. Mahdar, M. Ensuring the Quality of Electrical Power Supply. Berhad, National Standards Development Section, SIRIM, 2001.

28. NER directive on power quality. A regulatory framework for the management of power quality in South Africa. Rev.1.5, Pretoria, 2002.

29. Numerix-DSP. Algorithms, the Goertzel Algorithm. Digital Filter & Numerix Ltd. Siglab. Coalville, Leics. 2000.

30. Padiyar, K.R. HVDC Power Transmission Systems. Technology and System Interactions. John Wiley & Sons, New York. 1990.

31. Pansini, A.J. Electrical Distribution Energy. USA, McGraw Hill. New York. 1983.

32. Peregrinus, P. Power System Protection 2, systems and methods. The Electricity Council, UK, Stevenage. 1981.

33. Power Quality Network. Power System Training and Resources, Electrotek Concepts. USA, Knoxville, Tennessee. 1995.

34. PUTU. Programmable Universal Telecontrol Unit RTU Manual. Rev. 1. Telkor, Digital Systems Division, Pretoria. 1987.

35. Rationalized User Specification. Electricity Supply; Quality of Supply Minimum Standards for Application by the National Electricity Regulator. 1st ed., part 2, Cape Town. 1998.

36. Schnell, L, Sydenham, P.H. Technolgy of Electrical Measurements. University of South Australia, John Wiley and Sons, Chichester. 1993.

37. Sim Power Systems. Total Harmonic Distortion. The Math Works, Mathlab, Simulink, Stateflow, Handle Graphics, and Real-Time Workshop, Australia, Sydney. 2004.

38. Smith, S.W. The Scientist and Engineer's Guide to Digital Signal Processing. 2nd ed. California, Technical Publishing, Albuquerque. 1997.

39. Stanek, J. Micro Data Transmission design; Seminar Manual. Texas Instruments. Canada. 1996.

40. Stevenson, W.D. Elements of Power System Analysis. 3rd ed. North Carolina State University, McGraw Hill, Kogakusha. 1975.

41. Triangle Micro Works. DNP3 Overview. Rev. 1.1. North Carolina, Raleigh. 1997.

42. Van Wyk, L. Data Communications and Protocols for the ESI. The DNP3.0 Protocol. Astro Tech Projects and Systems engineering, Johannesburg. 2002.

43. Wadhwa, C.L. Electrical Power Systems. 2nd ed. University of Delhi, Wiley & Sons. New York.1991.

44. Whitaker, J.C. AC Power Systems. CRC Press, Florida. 1991.

OTHER LITERATURE CONSULTED

1. Ahmed, H; Spreadbury, P.J. Analogue and digital electronics for engineers. 2nd ed. New York, Cambridge University Press, 1984.

2. Arrillaga, J; Bradley, D.A; Bodger, P.S. Power system harmonics. New York, John Wiley & Sons, 1985.

3. Collins, R.A. The Giant Handbook of Electronic Circuits. Rev. 1.1, 1988.

4. Coney, R.G; Topham, G.H. The First SA Power Quality conference. Eskom, Power Quality Technology, South Africa, 1998.

5. DNP-3 Users Group guide. A forum for the supporters of the Distribution network Protocol. DNP V3.00 Subset Definitions. Ver. 2.00, 1995.

6. GEC Measurements. Protective Relays Application Guide. 3rd ed. The General Electric Company of England, 1987.

7. Groth, D; Bergersen, B; Catura-Houser, T. Network + Study Guide. A CompTIA Certification Program. San Francisco, Network Press, 1996.

8. Halsall, F. Data Communications, Computer Networks and OSI, 2nd ed. Addison-Wesley, 1988.

9. International Standards Organisation, Connection-Oriented Transport Protocol. DP 8073, 1983.

10. IST Training. ERTU & Bay Processor, Analogue input Card. Integrators of System Technology, Pretoria, 1994.

11. Jackson, L.B. Digital Filters and Signal Processing. Kluwer Academic Publishers, 1989.

12. Jaeger, R.C. Microelectronic circuit design. New York, Auburn University, McGraw-Hill, 1997.

13. Jayant, N. S; Noll P. Digital Coding of Waveforms, Principles and Applications to Speech and Video, Prentice-Hall, 1984.

14. Jordaan, G.D. Investigation into synchronisation for partial response signals and the development of a clock recovery scheme for 49QPRS signals. Bloemfontein, Technikon Free State, 1997.

15. Joseph, W.H. Applications of Discrete and Continuous Fourier Analysis. John Wiley & Sons, 1983.

16. Kraus, J.D. Electromagnetics, the McGraw-Hill series in Electrical Engineering, 3rd ed. McGraw Hill, 1997.

17. LAN's, WAN's, and the Internet. A+ Certification program. Rev. 1.3. Wave Technologies International, Inc. 1998.

18. Litter, J; Maher, J. Computers in the Laboratory. Longman Scientific & Technical, 1989.

19. Morton, A.H. Advanced Electrical Engineering. England, Longman Scientific & Technical Reprint, 1992.

20. Nakajima, E. The Sony Book of Digital Audio Technology. TAB Books Inc., 1983.

21. Oppenheim, A.V; Schafer, R.W. Discrete-Time Signal Processing. Prentice Hall Inc., 1989.

22. Papoulis, A.H Circuits and Systems, A Modern Approach. Rinehart and Winston Inc., 1980.

23. Peregrinus, P. Power System Protection 1, Principles and Components., UK, Stevenage, The Electricity Council, 1981.

24. Ravindranath, B; Chander, M. Power System Protection and Switchgear. New Delhi, Wiley Eastern Limited, 1994.

25. Roberson, D. Power System Protection Reference Manual. Reyrolle Protection. Boston, Oriel Press, 1982.

26. Roland-Mieszkowski, M. Introduction to Digital Recording Techniques, Proceedings from Acoustic Week in Canada, CAA Conference Canada, 1989.

27. Stallings, W. Data and Computer Communications. Macmillan, 1985.

28. Steingold, D.H. Analog-digital Conversion Handbook. 3rd ed. Englewood Cliffs, Prentice Hall, 1986.

29. Wannamaker, R; Lipshitz, S; Vanderkooy, J. Dithering to eliminate quantization distortion, the proceedings from Acoustic Week in Canada 1989 - CAA Conference, Halifax, N.S., Canada, 1989.

30. Wright, A; Christopoulos, C. Electrical Power System Protection. London, Chapman & Hall, 1993.

SOME INTERNET PAGES CONSULTED

1. Analog devices. ADSP-2181 EZ-KIT Lite Evaluation System Manual. 2001. www.analog.com/dsp.
2. Introducing the MODBUS Protocol. The Crompton Instruments Guide to RS485. Communications on the MODBUS Protocol. 2002. http://energy. Tycoelectronics.com.
3. Lee, J.G. The Effects of VSWR on Transmitted Power. 2002. http://www. webmaster/antennex.com.
4. Momentum Data Systems. Online Introduction to DSP. 2001. http://www.mds.com/training/on-line/intro.htm.
5. Numerix-DSP. Algrothms – The Goertzel Algorithm. Digital Filter + Numerix Ltd. Siglab v6. 2000. www.numerix-dsp.com/goertzel.html.
6. Professional Software Tools. Audio analysis and Processing. http://www. purebits.com/appnote19.html.
7. Texas instruments. DSP Developers Village. Development tools. http:// dspvillage.ti.com/docs/dspvillagehome.jhtml?DCMP=TIHeaderTracking&HQS= Other+OT+hdr_p_dsp.