Low bandwidth, image transmission amateur microsatellites

A.J. Mwambela

LOW BANDWIDTH, IMAGE TRANSMISSION ON AMATEUR MICROSATELLITES

A J. MWAMBELA B.Sc (Hons)

School Of Engineering And Computer Sciences

DURHAM UNIVERSITY, UK.

Thesis For Master Of Science submitted July 1993

ABSTRACT

Some recent amateur packet satellites carry open access digital store-and-forward transponders which implement common communication protocols known as PACSAT PROTOCOL SUITE. These standard protocols have improved a "friendly" interaction of different users of packet satellites throughout the world, hence, making packet satellites a more realistic means of communication. Application developments using packet satellites have resulted in an interesting electronic-mail network for medical applications, the Health-Net, where medical professionals in developing countries exchange information with their counterparts. The introduction of a higher rate of data transmission at 9600 baud rate compared to the traditional 1200 baud rate has improved the performance of these satellites. However, this new rate demands some modifications to the existing standard radio receivers and transmitters widely used. In particular, in view of the fact that, digital image technology has transformed microcomputers into powerful visual communication tools, this type of networks can be used for visual communications. Unfortunately, due to the orbit mechanics of satellites involved, the nature of communication protocols and the speed of data transmission currently available, transmission of image data through such networks is difficult in terms of transmission time. This thesis describes the application development of still-continuous tone image transmissions for visual communications, through such networks. It focuses on how to start a packet satellite transmission ground-station, and minimising the transmission time required for image data uploading and downloading, by compressing image data to remove visually insignificant data in the images. Image compression techniques, the internationally recognised JPEG compression technique and a novel compression technique based on FRACTAL, which are known to achieve higher compression ratios are used and compared in this work. Although expensive, FRACTAL compression technique has many advantages over the JPEG compression technique. However, owing to the cost effectiveness of the JPEG compression technique, it is recommended in this thesis for image compression application through Health-Net communication network.

DECLARATION

I hereby declare that the work in this thesis is entirely my own and that no part of it has previously been submitted for a degree in this or any other University.

Alfred J. Mwambela
(1993)

STATEMENT OF COPYRIGHT

Alfred J. Mwambela
(1993)

ACKNOWLEDGEMENT

I would like to take this opportunity to thank all people who in one way or another have participated in making this research work a success. I am grateful to my supervisor Dr. A. Purvis whose intellectual guidance, time, effort and patience he has given to encourage me in research, have helped me to produced this fruitful work. Also many thanks to my colleagues in research group whose constructive advice and friendship have been encouraging and their helpfulness has been much appreciated, P. Friend and his electronics workshop technical team, and all other staffs of the electronics department and school of engineering with whom the conducive research environment existing at the school rest on their shoulders.

I gratefully acknowledge the financial support given to me by Shell Oil company, FCO and the University of Durham.

CONTENTS

CHAPTER 3
PACSATS AND THEIR COMMUNICATION PROTOCOLS

CHAPTER 4
GETTING STARTED

CHAPTER FIVE
SATELLITE TRACKING

CHAPTER SIX
DATA DOWNLOADING AND UPLOADING

CHAPTER 9
CONCLUSION AND RECOMMENDATIONS

APPENDICES

GLOSSARY OF ABBREVIATIONS AND SYMBOLS

AFSK	Audio Frequency Shift Keying
AMSAT	Amateur Satellite Organisation
AMSAT-NA	Amateur Satellite Organisation- North America
AMSAT-UK	Amateur Satellite Organisation-United Kingdom
AOS	Acquisition Of Signal
ARS	Automatic Repeat Request
ASTRID	Automatic Satellite Telemetry Receiver and Information Decoder
BBS	Bulletin Board Services
BFT	Bath Fractal Transform
CCD	Charged Coupled Device
CCITT	Consultative Commettee Of The International Telephone And Telegraphy
CPE	Cosmic Particle Experiment
CW	Continuous-Wave Modulation
DCE	Data Communication Equipment
DCE	Digital Communication Experiment
DCT	Discrete CosineTransform
DE	De Emphasise Filter
DFT	Descrete Fourier Transform
DHT	Descrete Hartley Transform
DISC	Disconnect
DM	Disconnect Mode
DTE	Data Terminal Equipment
EIA	Electronics Industry Association
EIRP	Effective Isotropic Radiated Power
EIS	Earth Imaging System
FCS	Frame Check Sequence
FDCT	Forward Discrete CosineTransform
FM	Frequency modulation
FRMR	Frame Reject
FSK	Frequency Shift Keying

FTL0	File Transfer Level 0
GIF	Graphic Image Format
HDLC	High-Level Data Link Control
I	Information Frame
IDCT	Inverse Discrete CosineTransform
IFS	Iterated Function System
ISO	International Standards Organisation
ITT	Iterated Transform Technique
JPEG	Joint Photographics Experts Group
KISS	Keep It Simple Stupid
LABP	Link Access Procedures Balanced
LEO	Low-Altitude Earth Orbiting Satellites
LOS	Loss Of Signal
MOSFET	Metal Oxide Silicon FET
MPEG	Motion Picture Experts Group
NASA	National Auronatics And Space Administration, USA
NOAA	National Oceonic And Atmospheric Admninstration, USA
OSCAR	Orbiting Satellites Carrying Amateur Radio
PACSAT	Packet Satellite
PAD	Packet Assembler Disassembler
PBBS	Packet Bulletin Board Services
PBP	Packet Broadcast Protocol
PCE	Packet Communication Experiment
PE	Pre Emphasis Filter
PFH	Personal File Header
PID	Protocal Identifier
PSK	Phase Shift Keying
RADFET	Radiation Detection FET Transistor
REJ	Reject
RF	Radio Frequency
RNR	Receive Not Ready
RR	Receive Ready
RS	Radio Satellite
S	Supervisory Frame
SABM	Set Asynchronous Balanced Mode

SCTE	Solar Cell Technology Experiment
SSB	Single Side Band
SSID	Secondary Station Identifier
TAPR	Tuscon Amateur Packet Radio Corporation
TDE	Total Radiation-Dose Experiment
TIPE	Transputer Image Processing Experiment
TNC	Terminal Node Controller
TRDE	Total Radiation Dose Experiment
U	Unnumbered Frame
UA	Unnumbered Acknowledgement
UHF	Ultra High frequency
UI	Unconnected Frame
UOSAT	University Of Surrey Satellite
VHF	Very High Frequency
VLSI	Very Large Scale Integrateted Circuit
WHT	Walsh-Hadamard Transform

Chapter 1

INTRODUCTION

Computer data communication through voice graded communication channels has become common. By using existing telephone networks data can be routed from one place to another hundreds of miles away. For instance, in electronic mail systems a user places a message into a network, and the network relays the message from one node to another until it reaches its destination.

Similar developments have taken place in amateur radio communities. The telephone Bulletin Board Systems (BBS) have been used in microcomputer networking of individual or small user's groups through the existing public telephone networks. The Packet-radio Bulletin Board systems (PBBS) have been used in similar networks using radio waves as a medium of communication, sharing one bandwidth channel among many stations. The experience gained through PBBS operations has resulted in new developments of packet radio communications through amateur satellites.

Amateur satellites of the 1990s carry open access transponders. These communications transponders have made inexpensive, two way global communications possible in both real time, and through store-and-forward communications amongst radio amateurs throughout the world. Some new microsatellites such as OSCAR 22 are equipped with digital store-and-forward communication transponders. Spacecraft act like orbiting PBBSs where messages are stored on-board the spacecraft. Ground-stations can request that a message be read out on the downlink as the spacecraft passes near them. When a message is read out on the downlink it is not erased from the satellite memory. In this way, other stations can request a read out.

Digital store-and-forward communications transponders on-board packet satellites provide a cheap yet fast means of communication. A digital transponder is a non linear device optimised to receive digital data, process and re-transmit such data. The technique has been used in providing emergency communications for research activities in the Antarctic - an area which is not well covered by the current conventional geostationary satellites. More importantly, it has been used to provide 'electronic mail' communication for SateLife; a humanitarian organisation formed by 1985 Nobel Prize winner Dr. Bernard Lown to help medical professionals in developing countries easily to exchange information with their counterparts in the developed countries. SateLife uses OSCAR 14 for non-profit making electronic mail network for medical applications. Initially five Africa Medical schools have been using Health-Net to exchange electronic mail and receive up-to-date medical literature from different medical schools around the world. As a result of the successful experiment on the Health-Net network, a dedicated satellite for medical communications HEALTHSAT2 is scheduled for launch in 1993 [1].

The cost effectiveness of this type of communications is more attractive to developing countries where, due to limited resources and other priorities, it is difficult for the governments concerned to spend money on the present data communication networks. It is expected therefore, that this technique will find wide applications among the medical professionals in the developing countries and their counter parts in the developed countries.

Further, advanced technologies in digital image technology and its related fields have contributed in transforming microcomputers into visual communication tools. As a result, this has made possible the application development of microcomputers for digital processing of real-world pictures. Due to these technological advancements, the equipment for microcomputer visual tools is becoming cheaper, hence affordable to many individuals, schools and institutions. Although digital representation of images requires large disk spaces, the future trend indicates that real world pictures will be easily handled by microcomputers. Hence, visual

communications through store-and-forward electronic mail systems such as Health-Net will be economically attractive.

The objective of this project was to observe how useful the amateur microsatellites, together with their communication protocols, the PACSAT PROTOCOL SUITE, can be in distributing digital images through their digital store-and-forward satellite communications transponders. While normal text messages currently used may be adequate, pictorial presentation is more impressive. In order to improve the transmission time performance of this service for image data, large files such as image files need to be compressed. The project explores the possibilities of extending this application to medical image distribution via the described Health-Net program.

This thesis has been structured into nine chapters. The first chapter introduces the thesis by presenting the background and objectives of the project. Chapter two presents a short history of amateur satellites; their past, present and future trends and what can be expected from such satellite programs. The University of Surrey satellite program has been chosen as the model and is explained in detail. Chapter three describes the communication protocols, the PACSAT PROTOCOL SUITE, currently used by packet satellites and adopted in this project. This chapter also presents more details on protocols above the amateur X.25 protocols, the AX.25 link layer protocols; the Broadcast, the file transfer level 0 (FTL0) and personal file header (PFH) protocols which are more specific for LEO satellite operations, are presented.

Chapter four describes the work undertaken in starting a transmissions ground-station at Durham University for communications through 9600 baud FSK system satellites. An account of the general ground-terminal requirements; the antenna system, radio transceiver, TNC and computer requirements are presented. The choice of an antenna system for satellite operation and different experiments in improving the available antenna system are described. Modifications of an IC-47000 radio receiver and the

YAESU FT-2400H transceiver for satellite operations at 9600 baud rate and their performances are described.

Chapter five describes the method and the system used in tracking fast moving LEO satellites such OSCAR 22. Most amateur satellites of interest to this work are polar orbiting satellites. Knowing when they will be in range and where to point directional antenna is critical for efficient communications. Different experiments to confirm the accuracy of the predicted satellite passes are presented and discussed.

Chapter six presents and discusses the results of data uploading and downloading of message and image files carried out at this ground-station. Performances of the Broadcasting protocol and FTL0 protocols are discussed with reference to large files such as image files. When a packet satellite flies over an area of potential users, such as Europe, on any given pass, only a small percentage of users will be able to use the satellite because most satellites have a limited number of simultaneous users. As a result, large files such as image files require many satellite passes (a couple of days) to download or upload.

In order to improve the transmission time performance of this service for image data, large files such as image files need be compressed. Chapter seven present an overview of image compression. The irreversible compression techniques developed for still-continuous tone image compression are presented. The irreversible classical JPEG compression standards and new FRACTAL compression techniques which are known to achieve higher compression ratio are described. Chapter 8 compares the techniques and presents the result of compression obtained from the two methods and a suitable technique for Health-Net applications is recommended. Finally, chapter 9 concludes all the above experiments.

References

1. Sweeting N. M, 1992. *UOSAT Microsatellite Missions*.
Electronics and Communication Engineering J, pp 141-150, June.

Chapter 2

AMATEUR SATELLITES

2.1 Introduction

Amateur satellites are orbiting satellites carrying amateur band radio frequencies (OSCAR) for both the uplink and downlink of signals. Most of these satellites are non-governmental sponsored satellites developed, built and operated by scientists in universities, independent individual groups such as AMSAT-NA, and many others. This type of communications satellite provides a cheap yet efficient means of communication. Due to its cost effectiveness it is more attractive to universities and other independent organisations interested in learning space sciences and related subjects. This chapter gives a short account of the historical development of these satellites and the usefulness of such satellites, by describing the University Of Surrey satellite program as a model.

2.2 History Of Amateur Satellites

The first satellite to orbit the earth was SPUTNIK 1 in 1957. While this was the beginning of the space race by the former traditional 'super powers', it was also the beginning of the amateur satellite program. The success of this mission increased interest among radio amateurs in putting their own satellites into low orbit using radio frequencies for both uplink and downlink communications. The advantage of this was the application of low cost radio equipment (transceivers), which could be available to many individual users, in monitoring these satellites. However, the first amateur satellite OSCAR 1 was launched in 1961, four years after the SPUTNIK 1 launch. Since inception, amateur satellites have gone through three phases;

Phase 1: Early experimentation, OSCAR 1-5,

Phase 2:

Long life, LEO satellites plus state of the art experimentation, OSCAR 6 - 9, 11 & 12, 14 - 23 and RS 1-14,

Phase 3:

High altitude orbits with complex transponding and control systems, OSCAR 10 and 13. These have an advantage in that, being at a higher altitude than phase 2, they are usually available to the ground stations for about 8 hours a day which is good for both transmission and reception of data.

Phase 4:

Geostationary satellites are not yet launched and are still at the design stage and may go up to 2 GHz S-band away from the traditional 70 cm and 2 m bands.

More information about amateur satellites are contained in the book "The Satellite Experimenter's Handbook " [8].

2.3 Types Of Amateur Sattellites

There are two categories of amateur satellites based on the primary payload of all satellites, the communication transponder they are carrying. Those which use the real-time analogue transponders using mainly CW and SSB signals due to limited power available on spacecraft and those which use digital transponders using packets similar to terrestrial packet radio. A satellite that carriers a store-and-forward transponder designed so that it is relatively similar to terrestrial packet repeaters is frequently called a PACSAT. A linear transponder takes a slice of the RF spectrum that is centred about a particular frequency, amplifies the entire slice and re-transmits it, centred about a different frequency. When communicating

through a linear transponder standard CW-SSB transmitters and receivers for the uplink and the downlink are used.

On the other hand, a digital transponder is a non-linear device optimised to receive, process and re-transmit digital data. An example of such digital transponders includes the packet satellite designs, which are optimised for store-and-forward mailbox or bulletin board operation. Our interest is in the unit used on pacsat microsatellites, which are optimised for store-and-forward bulletin board and electronic mail activities. There are mainly two dominant types of store-and-forward digital transponders on-board packet satellites; the traditional 1200 baud PSK system and the new, fast, 9600 baud FSK system. For the purpose of this experiment only the fast 9600 baud FSK system, more attractive for image data transmissions, is considered.

2.4 Why Use Amateur Satellites

Amateur satellites have proven to be a cost effective way of learning space science and the related technologies [2]. While they are used in advancing space sciences, they have at the same time brought close the application of satellites to universities, schools and domestic users. It must be noted that satellite operations have traditionally been associated with high costs, a factor which has excluded the participation of such groups in using satellites and learning about space sciences. In order to understand the advantages associated with microsatellites a satellite program run by Surrey University is explained, hoping that it will shed light and provide a summary of what can be expected from such microsatellites.

The University of Surrey has pioneered microsatellite technologies since the beginning of its program. University of Surrey Satellite - UOSAT, is a local name given to amateur satellites operated by the University of Surrey. These are low altitude polar orbit satellites designed, built and operated in orbit by the University. The objectives of the program were stated as; (i) to investigate cost-effective satellite engineering techniques, (ii) to

demonstrate the feasibility and capability of microsatellites, (iii) to stimulate space education in schools, universities and the amateur radio community, and (iv) to catalyse the industrial exploitation of microsatellites [6]. Five satellites have been launched up to the time of writing of this report. The data transmission format of these spacecraft have been designed to be compatible with low cost receivers and PCs - to encourage direct participation of radio amateurs, scientists, universities and schools.

2.4.1 UOSAT 1

UOSAT 1 was the first in the series of satellites operated by the University of Surrey and was launched in 1981. It is known as OSCAR 9 in the amateur satellite series. It was designed to demonstrate the use of amateur microsatellites for space research and space education through numerous on-board experiments such as Geiger radiation detectors and computer-controlled speech synthesiser. Once in range bulletin news and operational status engineering telemetry broadcasts from the satellite could be received by any ground-station with proper receiving equipment. Over 3000 school children, university students and radio amateurs have participated directly in the mission [7].

2.4.2 UOSAT 2

UOSAT 2 is similar to UOSAT 1 in purpose and operation. It was launched into orbit in March, 1984 and continues to operate in a 700 km polar, sun synchronous orbit carrying a variety of payloads and experiments. Its name in the amateur satellite series is OSCAR 11. One of the most important experiments on board is the Digital Communication Experiment (DCE). The DCE was included on UOSAT 1 to provide in-orbit tests of radiation effects on VLSI components and to prove that store-and-forward communications using a low-cost satellite, and a small earth terminal was possible [3]. This experiment has pioneered store-and-forward mail-box communications using LEO satellites. A standard NBFM receiver is all that

is required to hear UOSAT 2 on 145.825 MHz in addition to a low cost decoder such as the ASTRID box, and a BBC or IBM-PC microcomputer to decode and display satellite data.

2.4.3 UOSAT 3

In amateur satellite series, UOSAT 3 is referred to as OSCAR 14. It was launched in January, 1990 into an 800 km polar, sun synchronous earth orbit. The primary payload carried by the UOSAT 3 is the Packet Communications experiment (PCE) which is an advanced digital store-and-forward communication transponder based on the results obtained in the DCE experiment on UOSAT 2 [4]. The PCE is supporting multi-access from hundreds of ground-stations and uses AX.25 packet radio communications protocols at 9600 baud rate transmitting on 435.070 MHz FSK. The PCE can be freely accessed by radio amateurs world-wide using inexpensive radio equipment, modems and IBM-PC microcomputers [1]. At the time of writing this report this spacecraft is only used for medical communications (Health-Net Network) and no services to radio amateurs are provided, all amateur services have been transferred to UOSAT 5.

Other non amateur application missions of PCE are to provide technical information and emergency communications for Antarctic research activities led by USA - an area which is not well covered by the current conventional geostationary satellites. It can also provide communications to different relief teams in areas which are poorly served by the existing data communications systems.

UOSAT 3 also carries two experimental payloads monitoring Cosmic Particles Experiment (CPE) and Total Radiation Dose Experiment (TRDE) to study the radiation environment experienced by satellites in low orbit and its effects on spacecraft semiconductor devices. The CPE, responds to cosmic particles as they pass through a large-area silicon PIN diode array by monitoring the charge they deposit using a charge integrator circuit monitored by an 80C31 microprocessor. Detailed analysis at the ground-

station can show how these particles are distributed around the orbit. The TDE provides a direct measurement of radiation absorbed at various points in a spacecraft by using radiation detecting transistors (RADFETs). Data from the CPE, DTE and housekeeping telemetry are transmitted in packets interspersed with PCE communications [5].

At the time of writing this report, the mission of the Packet Communications Experiment (PCE) is to provide global digital store-and forward 'electronic mail' communications for SateLife, an organisation formed by 1985 Nobel Prize winner Dr. Bernard Lown. SateLife is a humanitarian organisation formed to help medical professionals in the developing countries to easily and cost effectively interact with their counterparts in developed world. SateLife uses UOSAT 3 for a non profit electronic mail network for health professionals. Initially, five African medical schools, including one in Tanzania, use "Health-Net" to exchange electronic mail and receive up-to-date medical literature. Health-Net constitutes a direct application of store-and-forward satellite communications technique developed within the amateur services.

2.4.4 UOSAT 5

Unfortunately UOSAT 4 which was launched alongside UOSAT 3 lost communications with the commanding station one day after launch. UOSAT 5 or OSCAR 22 is the latest microsatellite launched by the University of Surrey. It was launched into a 775 km, polar, sun synchronous Earth orbit on July 23, 1991 [5]. The primary payload on UOSAT 5 is the Packet store-and-forward communications transponder which is the result of the two experiments explained above. Originally the mission for packet store-and-forward communications channel was to provide services for both radio amateur and non amateur users, but due to unavoidable circumstances it provides amateur radio services only.

It also carries other experiments: a Solar Cell Technology Experiment (SCTE) to monitor the performance of a range of indium Phosphide,

Gallium Arsenide and silicon solar cells during the life time of the microsatellite, to determine the effects of space radiation in orbit upon their long-term electrical performance; an Earth Imaging System (EIS) based on CCD camera to take images of the earth; and a Total Dose Radiation Experiment (TDRE) to enable researchers to determine the radiation dose absorbed by the SCTE and other VLSI components of interest located throughout the microsatellites [2].

2.5 Summary

Amateur satellites, through their digital store-and-forward transponders, have provided various institutions a more convenient way of learning space science independent of national governments, military and other international organisations which are traditionally associated with high cost. The success of the UOSAT program has generated much interest in exploiting the attractive attributes of microsatellites. The improvement of experimental payloads and experience gained are attracting sponsors interested in space science. More satellites are built by different institutes following similar steps. Different space-science related experiments, as demonstrated in these programs, can be performed at a reasonable cost. PACSATs, through their developed standard protocols, promise a more cost effective way of co-operation not only amongst radio amateurs throughout the world but also amongst scientists in the developed countries, and their counterparts in the developing countries as in Health-Net program. Through such a communication channel, an appropriate cost effective means of technology transfer which can help to narrow the existing gap, is born. The next chapter presents the standard communication protocols used in PACSAT satellites and will be used to communicate to the satellites.

References

1. Ward J, 1988. *The UOSAT-D Packet Communication Experiment.* ARRL 7th Computer Networking Conf. Proc., pp 186-193, Columbia,Maryland. October 1.
2. Sweeting M., 1992. *UOSAT Micro-satellite Missions.* Electronics and Communication Engineering Journal, pp 141-150, June.
3. Ward J and Price H, 1987. *The UOSAT-2 Digital Communications Experiment.*The Radio and Electronics Engineering, pp 57, September/October.
4. Ward J and Price H, 1986. *The UO-11 DCE Message Store-and-Forward System.* ARRL 5th Computer Networking Conference Proceedings, Orlando, US.
5. Sweeting M, et al, 1990. *University Of Surrey Launches Two New Satellites.* OSCAR NEWS, The Official J. Of AMSAT-UK For All Users Of Oscar Satellites, No. 81 pp 17-22, February.
6. Sweeting M, (edited by B.G Evans), 1987. *Cost-effective Space Engineering In Low Earth Orbit Satellites.*Satellite Communication System, Ch. 20 pp 359-375, Peter Peregrinus Ltd UK.
7. Underwood C. I , 1987. *UOSAT-2/OSCAR 11 Telemetry Data In School.* OSCAR NEWS, The Official J. Of AMSAT-UK For All Users Of Oscar Satellites, No. 65 pp 8-12, May.
8. Davidoff M, 1990. *The Satellite Experimenter's Handbook.* ARRL Publisher, Newington CT 06111 US.

Chapter 3

PACSAT SATELLITES AND THEIR COMMUNICATION PROTOCOLS

3.1 Introduction

The traditional terrestrial packet radio protocols have been found to be highly inefficient for LEO satellites operating store-and-forward PBBS [1]. New addressing schemes are proposed every few weeks, new routes and new ways of routing are proposed, tried, discarded and modified. As a result software maintenance on board these spacecraft becomes difficult. New and efficient standardised protocols suitable for packet satellite operation have been developed. This chapter describes the communication protocols, PACSAT PROTOCOL SUITE, currently in use in many packet satellites which has also been used in this project.

3.2 PACSAT

Amateur packet radio communications via a satellite is a recent development in amateur packet radio communications. The first to exploit packet radio was UOSAT 2 (OSCAR 11) in its DCE payload as explained earlier. Experience gained from this and other experiments has led via evolutionary steps to the present generation of packet satellites known as PACSATs. Pacsat is a generic term in the amateur radio services for LEO satellites which carry a store-and-forward digital transponder operating a standardised PBBS. All PACSATs use an advanced PBBS protocol called the PACSAT PROTOCOL SUITE [2].

3.3 Protocols For PACSATS

During the personal computer age, the telephone-line Bulletin Board System (BBS) and Packet Bulletin Board System (PBBS) have become common. Using a PBBS, messages stored on the PBBS by other users can be read and one station can store messages addressed to other users as well. Packet radio amateurs have developed their own protocol at the link layer known as AX.25 protocol. This protocol has been implemented in various PBBS systems for store-and-forward mail. Although this protocol works efficiently for terrestrial application, it has been found less suitable for communications with LEO satellites. Some of the grey areas which have been identified by engineers at Surrey in the early DCE and PCE experiments are;

. inability to transfer binary files,
. poor separation of message headers from the message data, and
. inability to support full-duplex data transfer [1].

As a result, new protocols operating above the AX.25 link layer have been developed. These protocols have been designed to overcome the above mentioned flaws and many others [2]. An important feature of this protocol is the ability to continue a 'transaction' on a later pass from the point at which you loose the satellite (LOS).

The success of these experiments and many others in different amateur satellites, has resulted in many packet satellites operating in a similar manner and are known as PACSAT. As a result, standardising communications protocols for the orbiting mail-boxes became essential. The engineers at AMSAT-NA and AMSAT-UK/UOSAT which are the main leading groups in amateur packet satellites developments, have agreed to use the common set of protocols on these new packet satellites. These protocols, generally known as PACSAT PROTOCOL SUITE, have been developed and are in application in many new PACSATs.

PACSAT PROTOCOL SUITE is implemented in several layers similar to the OSI-RM model layers. The OSI-RM model is the work of ISO in trying to standardise communications between computer systems. The OSI-RM model consist of a seven layer hierarchy (Fig. 3.1), which permits different computer systems to communicate with each other, as long as they adhere to the model. The first layer represents the lowest level and the seventh layer the highest level. Each layer is able to communicate only with the layers directly above and below it, and the interface between each layer is clearly defined [10].

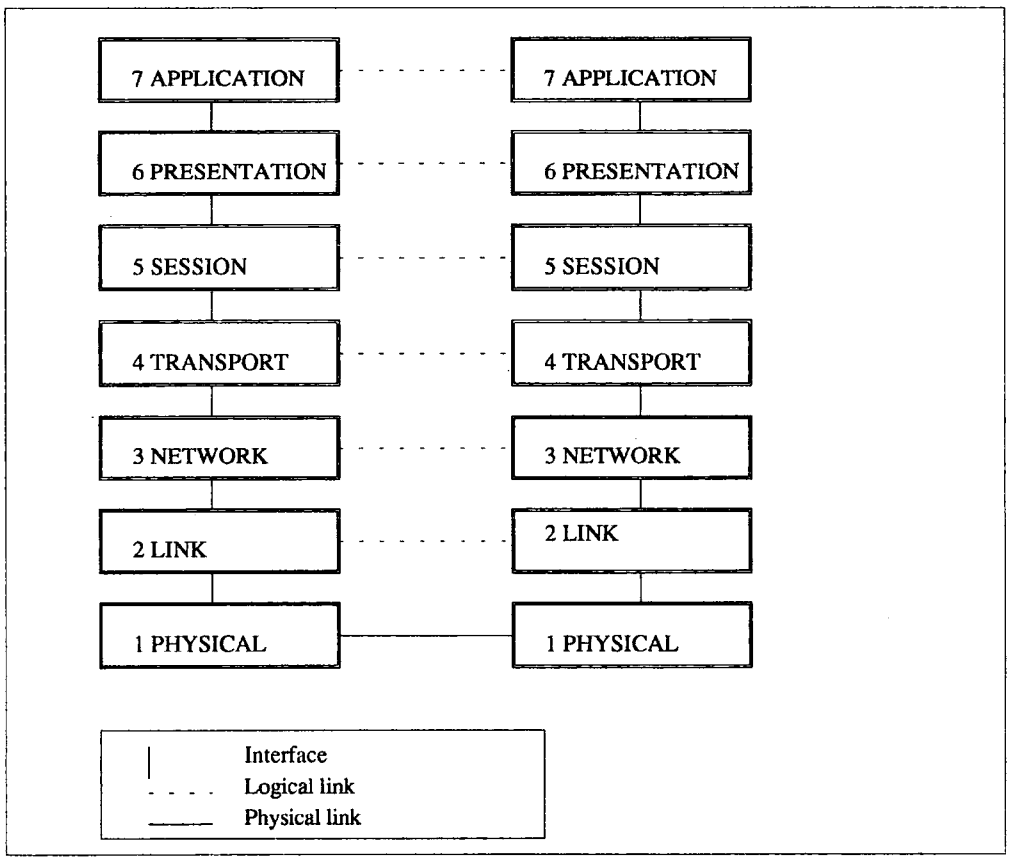

Fig. 3.1 - The ISO Open System Interconnection Reference Model (OSI-RM) for data communication network functions.

The Physical Layer has a role of physically (mechanically and electrically) moving data from device to device via the chosen medium of communication. For instance, the EIA (Electronic Industries

Association) standard physical layer for transfer of serial, and binary data between DTE (Data Terminal Equipment) and DCE (Data Communication Equipment) is the EIA-232D.

The second level, the Link Layer, has the task of partitioning data bits into frames and providing for the error free transfer of the frames to higher levels. Address information is added to each frame to facilitate the transfer from the source to a destination. ISO's High-level Data Link Control procedure (HDLC) is one standard that has been defined for the link layer.

The third level, the Network Layer, is concerned with routing frames through a network of links. This is accomplished by adding networking information to each frame. Two basic network layer approaches exist. A 'connection', or virtual-circuit protocol sets up and maintains a clearly defined path for all the packets transferred between the source and the destination during a single data communication session. A 'connectionless', or datagram protocol transfers each packet independently along the best available route. Hence, each packet must include complete addressing information.

The task of the fourth level, the Transport Layer, is to maintain a connection that is transparent to their source and destination by assuring that data received at the destination is complete and in the same sequence as the data that was sent from the source station. The fifth level, the Session Layer, provides data-flow synchronisation. The sixth level, the Presentation Layer, provides standards for the translation or interpretation of the data exchanged between the source and destination (for example, when the source and destination use different ways to represent the data). The seventh level, the Application Layer, provides an interface between the OSI-RM layers and the user application program (or programs) running on the computers.

In packet radio, the lower four levels have been significant in developing PBBS systems protocols. Protocols for physical and link layers based on

the CCITT recommendation X.25 protocols, are already in place and are known as AX.25. Some protocols for the Network and Transport layers are being developed [9]. It is above the AX.25 protocol where the PACSAT PROTOCOL SUITE start to differ from the traditional terrestrial protocols.

3.3.1 The AX.25 Protocol

At the link layer, packet radio amateurs have developed their standard protocol based on the ISO X.25 protocol. The Balanced Link Access Procedure (LAPB) of CCITT Recommended X.25, was used as the model for AX.25. X.25 protocol, is part of the OSI-RM for the standardisation of the interface to public packet switching networks. The amateur Packet-Radio Link Layer Protocol, version 2.0, commonly known as AX.25 protocol, is the amateur X.25 protocol. AX.25 specifies the contents and format of an amateur packet-radio frame and how that frame is handled at the link layer by packet radio stations. It comprises of three distinct and independent levels; the physical interface, the link control or link access protocol and the packet level protocol. The common protocols in these levels are RS-232C, HDLC and AX.25 Datagram or AX.25 Virtual-circuit modes protocols. These protocols have been implemented in all TNC-2 systems [11].

AX.25 works equally well in either half-duplex or full-duplex radio environments [9]. It is designed to work with connections between individual stations or between an individual packet-radio station and a multi-port controller. The primary differences between X.25 and AX.25 are that AX.25 accommodates amateur radio call signs for the addressing of each transmitted packet, and it provides the ability for unconnected stations to send packets permitting packet-radio operators to send beacon (by transmitting **UI** packet to identify the station) and to support datagrams at the link layer.

3.3.2 PACSAT Protocol Suite

The PACSAT PROTOCOL SUITE includes several protocols above the AX.25 protocols, which have been specifically designed for LEO satellites store-and-forward operations. Protocols above the AX.25 involved notably are; the Packet Broadcast (PB), File transfer Level 0 (FTL0) and Personal File Header (PFH) protocols. As already stated, the implementation of PACSAT PROTOCOL SUITE is in layers similar to OSI-RM model layers. A similar data communication protocol inter-connection is summarised in Fig. 3.3 [7]. At the lower levels AX.25 protocols are used, but at higher levels the above mentioned protocols are used. One of the outstanding PACSAT PROTOCOL SUITE features is that it applies non traditional packet radio access methods; the broadcast and file server methods.

In the broadcast access method, packets from the satellite are transmitted on point-to-multi point basis. Many ground-stations can simultaneously receive the transmitted packets without being necessarily connected to the satellite. In file server access method, protocols are implemented as client-server system where the satellite acts like a server for data storage and transfer and a ground-terminal as a client (Fig. 3.2) [7].

3.3.2.1 The Pacsat File Header Protocol

The PACSAT server can store and transfer data files but do not take any active roll in processing them. In order to ensure that these files can be properly identified and processed to the final destination, as in all E-mail messages, each file must contain some standard header items called PACSAT file header (PFH) [6]. These headers carries necessary information such as source-to-destination addresses, for efficient message delivery. The PFH protocol specifies types of headers to be included in the file for uploading or downloading. A proper program at the originator of the message adds PFH to the data file and at the final destination a similar program removes them to reveal the original

computer data. From Fig. 3.3, below the PFH envelope protocol, the PACSAT PROTOCOL SUITE stack divides to provide two message delivery protocols. The Broadcast protocol (PBP) which is point-to-multi point and the File Transfer Level 0 (FTL0) which is point-to-point protocols.

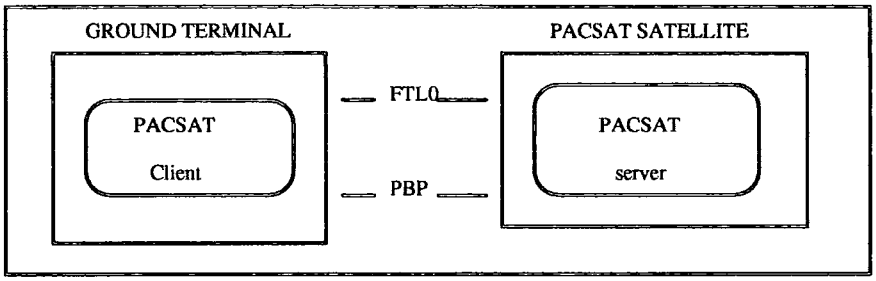

Fig. 3.2 Client-Server Block Diagram

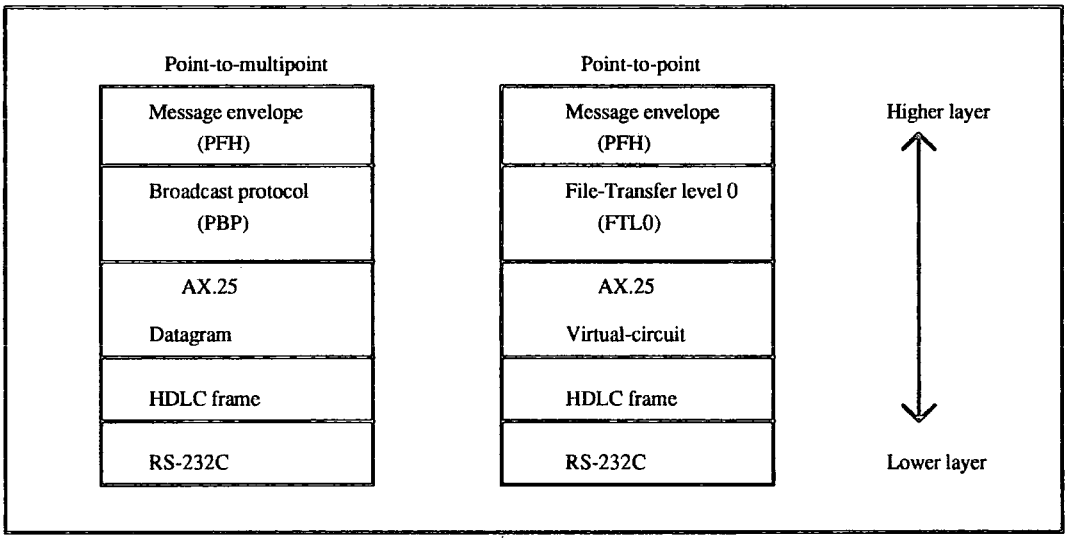

Fig. 3.3 Protocol Stack

3.3.2.2 Pacsat Broadcast Protocol

This is a message delivering protocol where a single information packet will be simultaneously received by all ground terminals in the satellite footprint. This is achieved by using the Datagram service of the AX.25 protocol which allows data to be transmitted from the source station

without a connection to the destination station. In this service a message to be transmitted by the PBP is first divided into a number of frames, each of which is broadcast on the UI frames. By using low-level error detection methods in the HDLC protocol, the ground terminal rejects any corrupted packets. The ground terminal only accepts each error free message packets and places the fragments in the proper location within the file to which it belongs [5]. Occasionally, due to local conditions, such as spin polarisation and 70 cm radar, which may cause signal loss and hence loss of frames, a ground terminal may miss some of the data packets, which leaves holes in the message. These holes are filled by a an error detecting routine known as selective ARQ (Automatic Transmission on Request) protocol, where terminal software sends a 'hole-fill' request to satellite and the satellite then sends only those packets which specifically fill the holes [7].

3.3.2.3 The File Transfer Level 0 Protocol

The FTL0 protocol is a point-to-point message delivery protocol used for message uploading and directory services. The operation of this protocol uses the error-free virtual-circuit mode of AX.25 similar to many terrestrial operations, although it includes some features which make it suitable for LEO satellite operations. The FLT0 protocol has the ability to continue data transaction on later passes from the satellite's LOS time. It also supports simultaneous bi-directional file transfer and multiple destination forwarding to make use of the limited satellite availability. These features are provided in as simple a manner as possible to avoid adding excessive overheads to the communication links and to permit implementation on a wide range of terrestrial microcomputers [4]. Unlike the broadcast protocol which provides its own error correction procedure, the FTL0 relies on the lower layer of AX.25 to provide an error free communication link.

3.4 Summary

The standard protocols developed will be used by all future packet satellites, speeding and bringing up high quality data communication to locations not currently served by the existing communication networks. Although there are some delays of data due to orbital mechanics in which LEO satellites are operating, the protocols provide a more reliable means of data communication very useful for applications where real-time communication is not important. The next chapter describes the work undertaken in starting a 9600 baud FSK system transmission ground-station at Durham University. This system uses the protocols described in this chapter to communicate through digital store-and-forward mailbox on board OSCAR 22 or OSCAR 23.

References

1. Ward J, 1989. *Protocols For Pacsats.*
   OSCAR NEWS, The Official J. Of AMSAT-UK For All Users Of
   OSCAR Satellites, No. 80 pp 2-8.
2. Price H. E and Ward J, 1990. *Pacsat Protocol Suite- An Overview.*
   ARRL 9th Computer Networking Conference, pp 203-206, London,
   Ontario, Canada, September, 22.
3. Price H. E Price and Ward J, 1990. *Pacsat Data Specification Standards.*
   ARRL 9th Computer Networking Conference, pp 207-208, London,
   Ontario, Canada, September, 22.
4. Price H. E and Ward J, 1990. *Pacsat Protocols: File Transfer Level 0.*
   ARRL 9th Computer Networking Conference, pp 209-231, London,
   Ontario, Canada, September, 22.
5. Price H. E and Ward J, 1990. *Pacsat Broadcast Protocol.*
   ARRL 9th Computer Networking Conference, pp 232-244, London,
   Ontario, Canada, September, 22.
6. Ward J and Price H. E, 1990. *Pacsat File Header Definition.*
   ARRL 9th Computer Networking Conference, pp 245-252, London,
   Ontario, Canada, September, 22.
7. Ward J, 1991. *Microsatellites For Global Electronic Mail Network.*
   Electronics and Engineering Journal, pp 267-272, December.
8. Bleazard G. B, 1982. *Handbook Of Data Communication.*
   NCC Publisher, 2ED.
9. Horzepa S, 1989. *Your Gateway To Packet Radio.*
   ARRL Publisher, Newington, CT 06111 USA.
10. Hanshall J and Shaw S, 1988. *OSI Explained-End To End Computer*
    Communication Standards.

Chapter 4

STARTING A PACSAT GROUND-STATION

4.1 Introduction

Starting an amateur packet satellite communications ground-station for satellites operating with a 9600 baud FSK system requires a certain minimum number of equipment. Some of the equipment needs modification to operate efficiently with data transmission at 9600 baud rate. This chapter presents work done to start such a ground-station and the necessary modifications made to some equipment to improve its performance, including the choice of an antenna system, modification of a receiver and a transmitter available, and specification of other requirements.

4.2 Getting Started

In order to start an amateur packet satellite receiving ground-station, some decisions have to be made about the type of packet satellite of interest. Since the main interest is in the application of store-and-forward network for image transmission, the fast 9600 baud FSK system packet satellites are used. These satellites use uplink and downlink signals using a Mode-J (70 cm downlink / 2 m uplink) band plan. The choice of this mode is based on the experience gained in the previous DCE experiment on UOSAT 2 which showed 70 cm to be significantly quieter at the ground-station than the 2 m [24].

In order to communicate with these satellites, there are minimum requirements of both hardware and software at the ground-station [23]. This is due to the fact that the satellite acts like a file server and the PACSAT

protocol suite is implemented as client - server system in which the satellite acts as a server, and software in the ground terminal acts as a client. At the time of writing this report, there only two freely accessible 9600 baud FSK system satellites, OSCAR 22 and OSCAR 23.

4.3 Hardware Requirement

The minimum basic hardware requirement for reception of broadcast from satellites with 9600 baud FSK system is summarised in the diagram below (Fig. 4.1). A similar arrangement can be used for 1200 baud PSK system where an FMn receiver is replaced by an SSB receiver and an FM transmitter by an SSB transmitter. The complexity of the ground-station depends on the purchasing power of the user, and his intended use of the satellite. For instance, if someone is interested in receiving the broadcast bulletin news only, a transmitter is not necessary. However, for a serious user of this satellite, a more complex or automated station, is required.

4.3.1 Choice Of Antenna System

The primary duty of a receiving aerial is to obtain as much signal power as possible from the required wave. Some aerials are required to work at high efficiency over a relatively wide range of frequencies; others are designed, essentially, to respond over a relatively narrow band of frequencies. The latter type is widely used in amateur radio, hence amateur satellites as well. Therefore, the choice of ground-station antenna will depend on the uplink and downlink frequency band plan mode of the satellite under consideration. In this case, OSCAR 22 and OSCAR 23 use mode-J band plan, hence any VHF / UHF antenna is suitable. The electrical performance of an antenna is determined by many factors such as the beam shape, polarisation, efficiency, input matching, and so on [22]. Although there is no intrinsic difference between antennas for satellite work and terrestrial work, some types of antenna work well for satellite link than the others.

Properties that make certain types of antenna desirable for terrestrial operations may make it a poor performer on a satellite link.

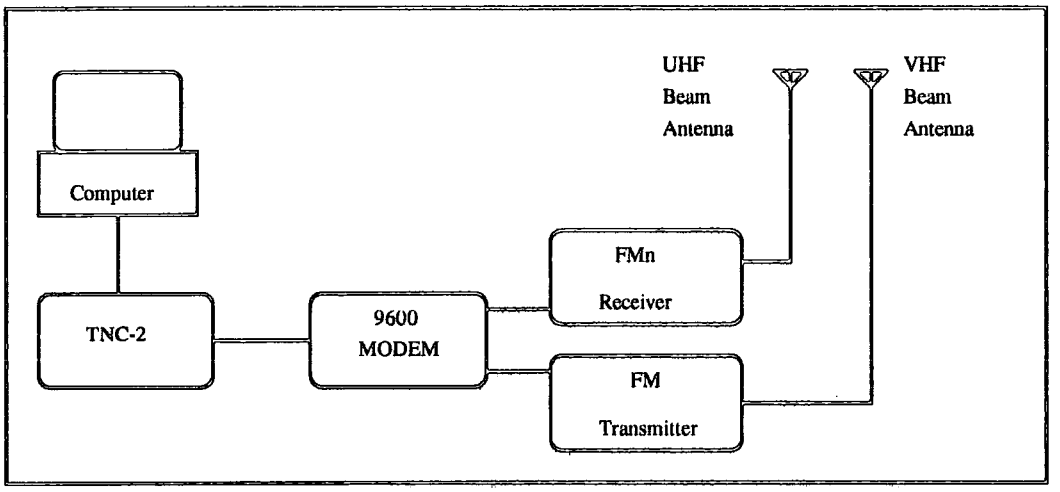

Fig 4.1 Ground-station Basic Hardware Requirement

4.3.1.1 Polarisation

Signals arriving from most LEO satellites are elliptically polarised due to spin and changing satellite altitude, and partially to faraday rotation imparted by the ionosphere as the signal passes through it. It has been found that a circular polarised antenna at the ground-station produces best performance whether a signal is linearly or circularly polarised [4]. Hence, a circular polarised antenna will produce best results when receiving these elliptically polarised signals from the satellite. By the same argument, since the receiving antenna properties are the same for transmitting antenna (reciprocity theorem), a circular polarised antenna at a ground-station will provide optimal result on the uplink.

4.3.1.2 Gain Pattern And Beam Width

Due to the orbit mechanics of LEO satellites such as OSCAR 22, the gain pattern of an antenna may also affect its performance in satellite links. One

important characteristic of the gain pattern is the beam width. High antenna gains are obtained by concentrating the radiated power in a specific direction. High gains can only be obtained by sacrificing beam width. Therefore, a beam antenna with high gain is a more convenient antenna to use in LEO satellite links to give adequate downlink S/N ratio and cost effective desired EIRP. However, due to its associated narrow beam width, it is necessary to find a convenient way of accurately aiming the beam as the satellite passes by.

Beam antenna are considered best for amateur satellite links. Several types of beam antennas in the market today are recommended for amateur satellite links, for example Yagi, Quad, Loop, and Helix antennas. The performance of these antennas is similar in terms of pattern and ground effects, but each has its own problems in terms of difficulty in construction, mounting ease, commercial availability and suitability for use as part of a circularly polarised system. Most commercial manufactures favour the Yagi over the other types of beam at 2 m and 70 cm because it has a simple structure, is light weight, and has a low wind load for a given gain [19, 20, 21]. The available antenna system at Durham University ground-station (at 2 m and 70 cm) is a commercially constructed VHF/UHF crossed Yagi antenna system.

At UHF/VHF frequencies the loss associated with feed-lines is critical. Low-loss cables are needed to avoid significant loss of signal strength before reaching or leaving the radio. The recommended cables for transmission line between the antenna and radio are the RG-8 / RG-213 coaxial cables for short run (less than 25 feet). A low-loss cable such as Belden 9913 is more suitable for both short and long runs. The recommended connectors for use in this set up are the N-type connectors [23].

4.3.1.3 The Antenna System Performance

The performance of the UHF antenna which is used for signal downlink, was not as good as the theoretical expectation. For instance, the wide band DISCONE antenna also available at this ground-station, produced a better result compared to the UHF Yagi antenna, during the satellite overhead passes. This is contrary to the theoretical expectation, where an 18 elements UHF Yagi antenna with such a gain is supposed to produce better performance compared to the DISCONE antenna. Two cables of different quality have been used for the two antenna systems, the better one being used in DISCONE antenna. This suggests that there is signal loss between the UHF antenna and the radio receiver. Further measurements on the strength of the arriving signal using a spectrum analyser revealed that there is a difference of about 10 dB between signals from the two antennas, with the Yagi antenna having small gain. Further tests on the antenna where a transmitter was used to transmit on the UHF frequency showed that the antenna was behaving as expected. Therefore, the coaxial cable used in the UHF antenna was thought to be the source of signal loss. The most appropriate and quickest solution to these problems was to install a pre-amplifier.

A MICROSET GaAs UHF (430-440 MHz) pre-amplifier with a maximum gain of 15 dB was installed on the UHF antenna system. This installation improved the performance of the UHF antenna system in monitoring the downlink signals from OSCAR 22 and OSCAR 23. The number of satellite passes which could be received by the ground-station has increased from one, which was the overhead pass, to at least four good passes. This implies that the access time to the satellite has increased from approximately 10 minutes to 40 minutes in 12 hours. The VHF antenna system which was used for uplink, was tested and found to work to the theoretical expectations. This system has been successfully used to log into the computer server on board OSCAR 22.

4.3.2 Radio Equipment And Their Modifications

The quality of the entire radio link is dependent on the radio equipment. Data transmission using the 9600 baud FSK system is a departure from the traditional 1200 baud AFSK system widely used in amateur packet radio. Although this system increases the amount of data transferred per second, it imposes some limitations to the existing voice graded radio link. The 1200 baud AFSK system uses audio tone frequencies (1200 and 2100 Hz) FM modulated onto a carrier which make it easy for data to pass uncorrupted through the bandwidth and audio filters found in many radio equipment. In contrast, the 9600 baud FSK system uses shifts between two radio frequencies, say 145.009850 and 145.10150 MHz for nominal operating frequency of 145.01 MHz. As a result, this fast system requires a slightly wider bandwidth and incurs an increased amount of system noise which cannot be overcome by the existing audio filters, and they must be bypassed. Therefore, not all radio equipment operating with the 1200 baud AFSK system will be suitable for data at 9600 bps .

In order to receive broadcast from 9600 baud FSK system satellites, an FM radio receiver capable of receiving at UHF frequencies is needed. For the best results it should have an AFC to track the Doppler-shift present on the signal or a wide IF is recommended. A NBFM receiver with a typical bandwidth between 15 - 20 MHz is preferred. The FM receiver should be modified so that the audio signal can be tapped from just after the demodulator to avoid any audio filtering [2]. Similarly, in order to transmit to the satellite, an FM transmitter capable of transmitting at VHF frequencies is needed. Its transmitting power depends on the desired EIRP for the satellite under consideration. For OSCAR 22 it should be able to produce 100 W EIRP. Not all FM transmitters can be used for data transmission at 9600 bps. It has been found that only true FM transmitters whose FM is generated by direct method works well with data at 9600 bps [17].

Ordinary radio receivers and transmitters have been used in data communication using voice graded radio-link channels. These devices were

designed for voice rather than data communications. Whilst they can accommodate data in their normal operation using the 1200 baud AFSK system, they introduce errors in higher data rates. Pre-emphasis and De-emphasis audio filters (PDF) present in many radios (Fig. 4.3) normally introduce distortion such as group delay distortion at 9600 bps, and must be bypassed. PDF technique is a filtering technique widely applied in FM transceivers intended to increase the S/N ratio at the audio output by boosting the most noise-susceptible portion of the signal before noise becomes a problems (at the transmitter) and then invert the process at the receiver [18].

At the transmitter, where the signal levels are high and noise is, to some extent, under control, the modulating signal, is purposely altered with the aim of enhancing the higher frequencies relative to the lower ones. This tailoring of the signal produces distortion, and the receiver must therefore invert the process to furnish an undistorted version of the modulating signal. The initial alteration of the signal that produces high-frequency enhancement is known as Pre-emphasis filtering; the inverse process is De-emphasis filtering. The De-emphasis filter enhances the lower frequencies relative to the higher ones. A typical De-emphasis filter network is as shown in Fig 4.3.B. The net gain is that the noise now adds to a strong high-frequency signal and is suppressed in the de-emphasis filtering.

Like all filters, PDF shifts phase of the signals under consideration, and is more apparent when approaching pass band edges. This shift in phase is more significant at the higher rate of 9600 bps. As a result, component frequencies are transmitted with varying velocity, the received phase relationships will differ from the sent phase relationships resulting in distortion, known as group delay distortion, a form of distortion which is one of the major limiting factors in data transmission using voice graded channels. Whilst the effect of delay distortion of few milliseconds is not significant in voice signals, it severely impairs performance of the data transmission radio link. A former digital 1 in the 112^{th} position in the wave train can be delayed to where the 113^{th} position should be etc., and the data is said to corrupt. Therefore, a direct connection to the FM modulator

varactor and FM detector is needed for the G3RUH format 9600 baud packet.

4.3.2.1 Radio Receiver Modification

The performance of FM receivers and their advantages have long been subject in radio communications [11]. A block diagram of a typical FM receiver is shown in Fig. 4.3A. For a normal radio receiver to accommodate data rate at 9600 bps, the DE filter is by-passed by tapping the audio signal as it leaves the discriminator. A typical DE filter network is as shown in Fig 4.3B. The transfer function for the DE filter network is given by eqn.(4.2). The designed receiver break frequency is usually 2.1 kHz when $\tau_1 = 75\mu s$ which is acceptable for voice and data at 1200 bps AFSK.

$$T(\omega) = 1/(1 + j\omega\tau_1) \quad \text{.....(4.2)}$$

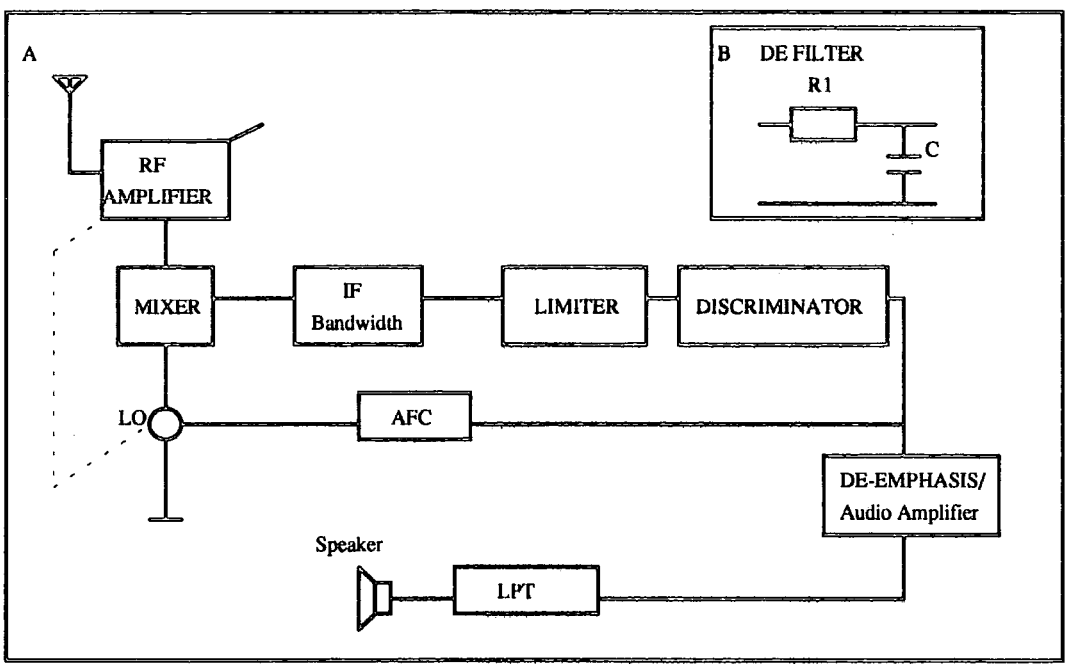

Fig. 4.3 Block Diagram Of An FM Receiver

The radio receiver available at this station is the IC-R7000 scanner designed for a variety of users covering HF, VHF and UHF bands. It incorporates an AM mode, FM mode using both narrow and wide filters, and an SSB mode both USB and LSB selection which make it very useful for amateur satellite application. Like many old receivers, its discriminator circuit is followed by audio filters and amplifiers (Fig. 4.6). Hence a direct FM detector signal output is required to work with the G3RUH 9600 baud modem. This scanner has been successfully used to receive data at 9600 bps and has been found to perform well by Branegan [6]. Referring to the circuit diagram of the scanner (Fig. 4.7), unfiltered audio signals are tapped after the discriminator on the IF unit at the common point of R97 and R96. A shielded coax has been used to connect this point to a spare position on the rear back plate of the receiver.

Interfacing details between a radio receiver and a TNC have been well covered in the PacComm TINY-2 TNC operating manual [5]. However, at this point the tapped audio signals have unwanted component frequencies detected together with the data signals and must be removed before feeding the signals to a 9600 baud modem. An external modified low-pass filter similar to a De-emphasis filter to increase the break frequency (eqn.(4.2)) has been added (Fig. 4.4 A) with C = 10 pF and VR2 = 10k.

Due to this modification, there is a circuit impedance mismatch between the receiver and a TNC circuits. The receiver has high impedance compared to the set TINY-2 NB-96 TNC normal impedance. As a result of this, data are corrupted before reaching the modem. A normal impedance mode setting of the TNC has been changed to the higher mode and data can now reach the modem to the final computer destination uncorrupted. After these modifications the receiver has been tested and found to function satisfactorily with the 9600 baud FSK system. It has been used to receive broadcast data from OSCAR 22 and OSCAR 23 (appendix 2) in the current experiment without problems.

4.3.2.2 Transmitter Modification

Not all FM transmitters are suitable for data transmission at 9600 bps. Two main limiting factors affect the performance of FM transmitters at 9600 bps, the bandwidth limit available and method used in generating FM. The 9600 baud FSK system requires a slightly wider bandwidth of at least 15 kHz. There are two methods of generating FM - by a direct method sometimes referred to as Crosby or true FM, and the indirect method sometimes referred to as Armstrong FM. Only true FM transmitters have been found to work with the 9600 baud FSK system [17]. Armstrong FM transmitters generate FM by changing the phase of the crystal oscillator's output, as a result they are bound to cause group delay distortions.

Similarly, for a direct FM transmission system to work with data at 9600 bps, assuming the system has a wide enough bandwidth, the audio PE filter must be bypassed. There are different points for injecting the modulating signal from the TNC for different transmitters. Most synthesised rigs can be made to work, with only simple modifications. For true FM rigs, only a couple of internal connections and de-coupling components are required to bring in the modulating signal directly to the FM modulator input.

The transmitter available at this station is the YAESU FT-2400H transceiver. Although the rig is modern, its microphone jack is pre-emphasised. Therefore, G3RUH 9600 baud modem cannot be directly interfaced with the transmitter through its microphone jack. For YAESU FT-2400H the input to the modulator is pin 4 of J1002 connector in the VCO unit (Fig. 4.8 and 4.9). The generation of FM in this transmitter is by direct method whose modulation is by variable reactance method [8]. This is achieved by using a tuned circuit oscillator in which one of the resonant circuit elements is a variable reactance in the VCO unit.

Detail on how to connect a transmitter to TNC can be found in the Tiny-2 PacComm operating manual [5]. As it turns out, the microphone impedance of this rig is higher (800Ω) than a TNC designed maximum impedance value of 500Ω, and the drive level of a 9600 baud modem is higher at the

input of the modulator. A TA signal from pin 1 of the TNC's five pin DIN connector is connected to the VCO modulator input pin 4 of J1002 connector via a de-coupling circuit (Fig. 4.4B). The TA signal is injected through two 22 μF capacitors connected back to back to form a non-polarised capacitor of about C = 10 μF through a VR3 = 22k variable resistor into the modulator [16]. The TA signal is adjusted using the variable resistor VR1 on the TNC and VR3 such that at most a 60 mV peak to peak signal appears at the modulator input. This will ensure that the deviation produced at the output will not cause the PLL circuit to lock during transmission of a reasonable size data file. A complete modified transceiver system has been summarised in Fig. 4.5.

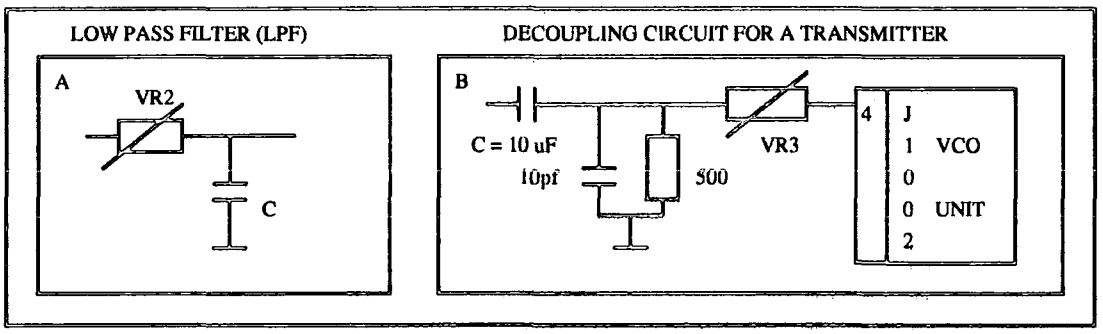

Fig. 4.4 A: Coupling Circuit For IC-R700 Receiver And a TNC, B: Coupling Circuit For The YAESU FT-2400H transmitter And a TNC.

The transmitter was tested and found to work with some limitations, at 9600 baud rate. Only small files of about 500 byte can be uploaded to the satellite using this system (appendix 3). For large files, the transmitter will transmit a portion of the file and stop. This is due to the fact that it has a narrower bandwidth (10 kHz) compared to the recommended bandwidth (15 kHz) for 9600 baud rate operation and a PLL circuit, usually a very narrow band filter, which tends to stop modulation of the carrier every time the set frequency deviation limits are exceeded [17]. As a result together with limited satellite server accessing time, big files cannot be transmitted to the server using this transmitter. Widening the PLL filter is beyond the scope of this work and may need further consultation with the manufacturer. However, a transmitter with this modification can be used for

data downloading operations, where ARQ (Automatic Retransmission on Request) frames sent by the station to the server are less than 500 bytes, typically 244 bytes.

4.3.3 Modem

The original design objective and the analogue technology employed in voice graded communication equipment, imposes some constraints on transmission of data originating in digital form. In order to transmit digital signals along analogue circuits, it is therefore necessary first to convert them to a form which is acceptable to the circuits which are provided. This is the role of a modem, whose chief functions according to Sherman [14] are: to match the channel bandwidth provided; to provide information within the transmitted signal stream for synchronisation purposes; to modulate the signal onto higher frequency carrier; and finally to compensate for a variety of other factors which might impair the quality of transmission.

The G3RUH 9600 baud modem is a typical example of such modems. There are different modems for 9600 baud packet format in the market today: G3RUH 9600, PacComm NB-96, Katronics DE 9600, Tasco TMB-965, K9NG or similar modems [2]. These devices will augment the modulator in the TNC modem on transmit, replace the demodulator in the TNC modem on receive, and provide the signal needed to enable a receiver to track the satellite downlink frequencies as they shift due to Doppler effect. A TNC available at this station is the PacComm TINY-2 NB-9600 a high speed version of TINY-2 with a built in G3RUH 9600 baud modem. This modem is a complete TNC-2 firmware compatible, supporting all TAPR (Tucson Amateur Packet Radio Corporation) commands and many other additions and personal message system.

4.3.4 Terminal Node Controller

A TNC with internal or external 9600 baud modem will allow a computer and radio to converse with the computer aboard the 9600 baud FSK system satellites. It is in a TNC where the lower layers of the PACSAT protocol suite, as in the OSI-RM model; the RS-232C physical and AX.25 packet radio network link layers are implemented. It has a PAD (Packet Assembler Disassembler) computer circuit which assembles packets. It also disassembles packets accepting frames received, extract data from the packet frames, and transferring the data to a terminal. The TNC should be operating in KISS (Keep It Simple Stupid) mode. In this mode the TNC does a bare minimum of processing and passes the data, almost raw, through to the computer attached to its serial port [11]. For the purpose of this project, a TNC must have either an internal 9600 baud modem or an external connection to support the 9600 baud modem. A TNC required for this operation should be completely TNC-2 firmware compatible, which support all TAPR commands. These conditions are pre-conditions for using the ground-station software PB and PG from the University of Surrey.

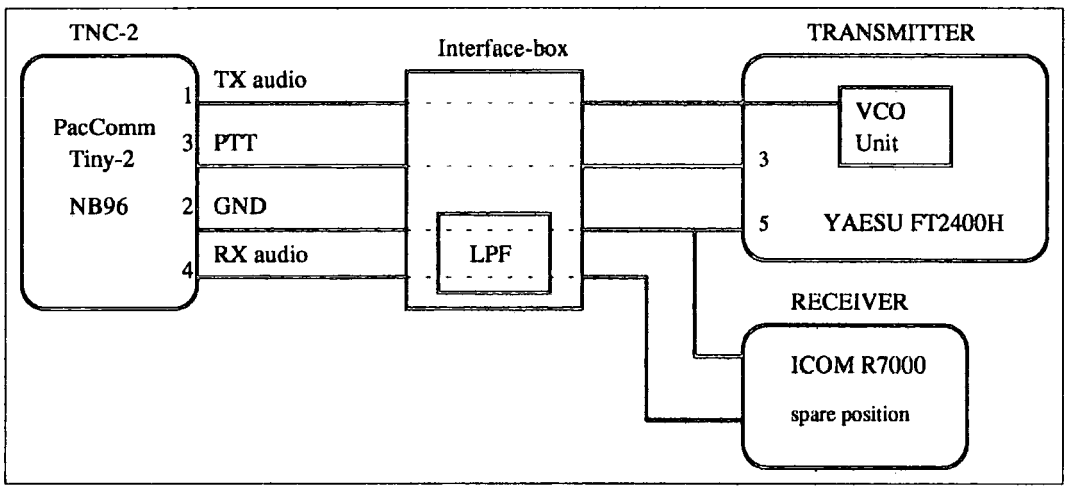

Fig. 4.5 Modified Interface Connection Between The TNC And Radio Equipment.

4.3.5 The Computer

The choice of a computer type is limited by the availability of software implementing the PACSAT protocol suite for different brands of computers. At the moment IBM clones, MAC or an Atari ST can be used. Other machines are likely to be catered for as soon as software writers implement the pacsat protocols [2]. For OSCAR 22 and other packet satellites which use the 9600 baud FSK system, any IBM clone is suitable as long as its clock speed is at least 12 MHz, and has a RAM disc or Hard disc memory. The available computer at the station where this work was undertaken is the RM NIMBUS VX/2 computer which is a 386 PC with a 12 MHz clock. However, experience shows that a faster machine is useful.

4.4 Software Requirement

The Software needed for PACSAT satellites are available from a number of sources and for different computer families. For the purpose of this project, the "official" University of Surrey software was used. It contains two parts: PB to access the Broadcast system, and PG to allow uploading of files and various other tasks. The software available for the IBM clones which have hard disks, and run MS-DOS from the AMSAT-UK office are;

. PG.EXE - FTL0 Protocol,
. PB.EXE - Broadcast Protocol,
. PFHADD.EXE - File Header Add utility, and
. PHS.EXE - Header Striper.

PG.EXE is the program for accessing satellites which uses the PACSAT protocol suite. PG implements the connect-mode file transfer protocol, the FTL0 protocol. At the time of writing this report, this program had been modified so that it could only be used for uploading files to the satellite's servers [23].

PB.EXE is a ground-station program for capturing broadcast from PACSATS. It is used to receive messages and message directories from packet satellites. At the time of this report the new version of PB.EXE included a PHS utility. The PHS.EXE program is no longer required to process completely downloaded files in a format which can be easily read and is no longer supplied as a separate program by AMSAT-UK. The function of PHS.EXE was to display PACSAT file headers, and remove headers from files which have been downloaded from PACSATs.

PFHADD.EXE program prepares files for uploading. Only files which have been prepared by PFHADD.EXE may be uploaded to PACSATs. More information about PB and PG programs can be obtained from "the PG and PB Users documents" [9].

4.5 Summary

This chapter discussed how an amateur satellite ground-station for 9600 baud FSK system satellites can be set up using the available equipment and their modifications. However, the YAESU FT-2400H transceiver generally cannot be used for data transmission at 9600 baud rate with the discussed modification. This transceiver can be modified to work with data at 9600 baud rate by widening the bandwidth, hence the PLL filter, if the manufacturer is consulted and can co-operate. Although a high gain UHF antenna may be useful in downlink communication, it is important to add a pre-amplifier to the antenna system to increase the efficiency of signal reception within a short duration of the satellite pass. Due to Orbital mechanics of LEO satellites and the nature of beam antenna used, a more accurate means of tracking the satellite in question is important. The next chapter presents means and ways of tracking phase II amateur satellites.

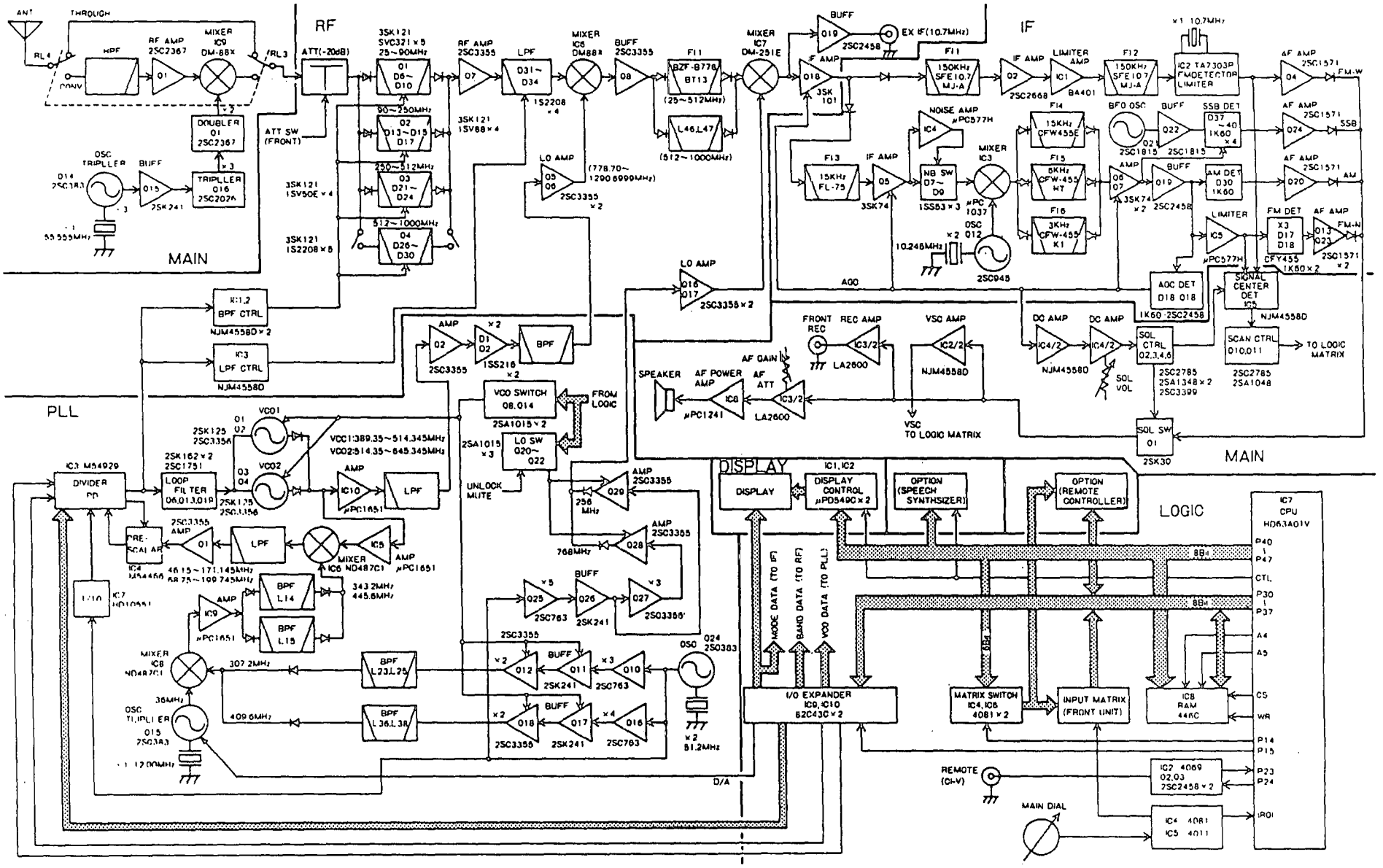

Fig. 4.6 Block Diagram Of IC-R7000 Receiver

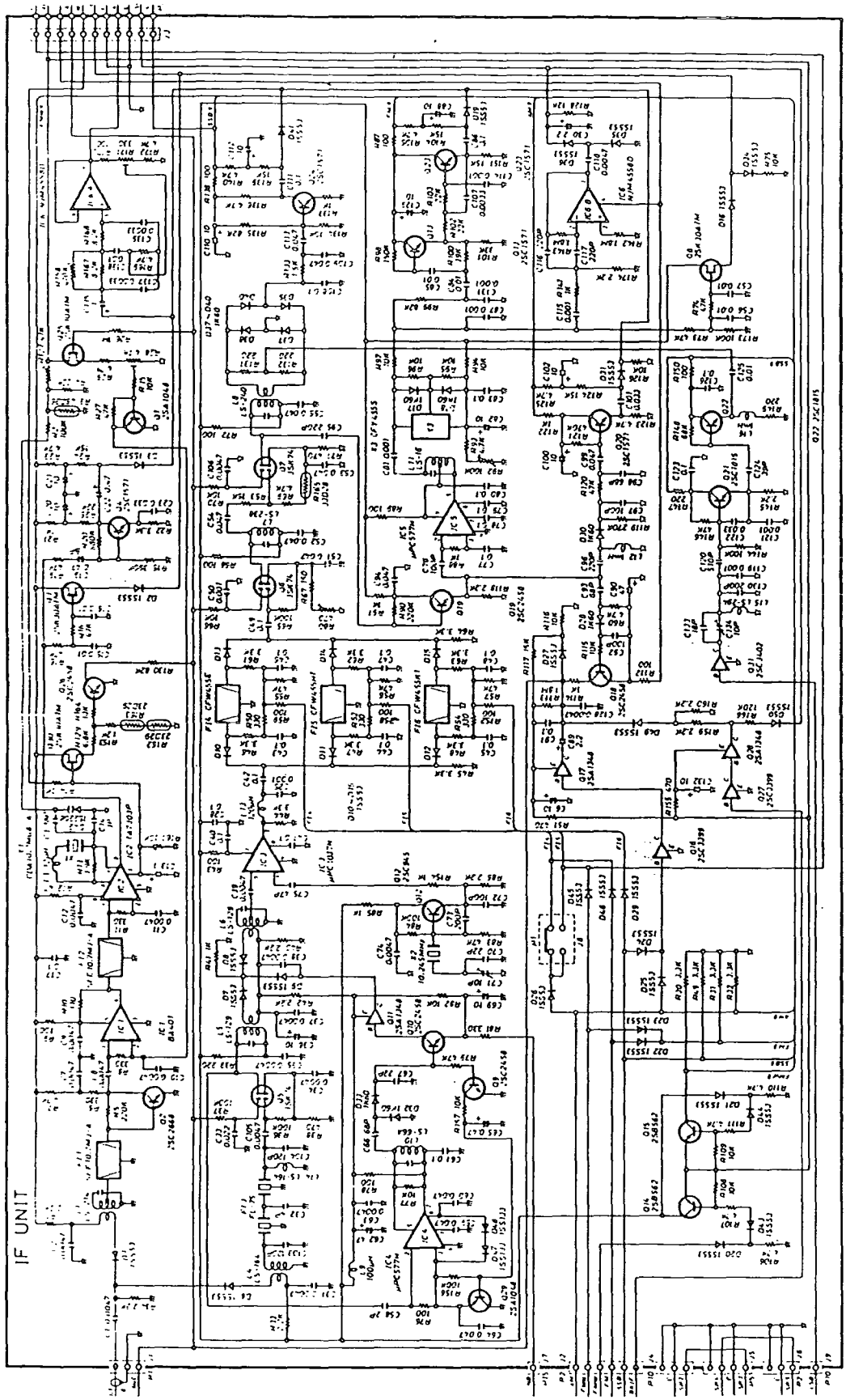

Fig. 4.7 IC-R7000 IF Stage Circuit Diagram

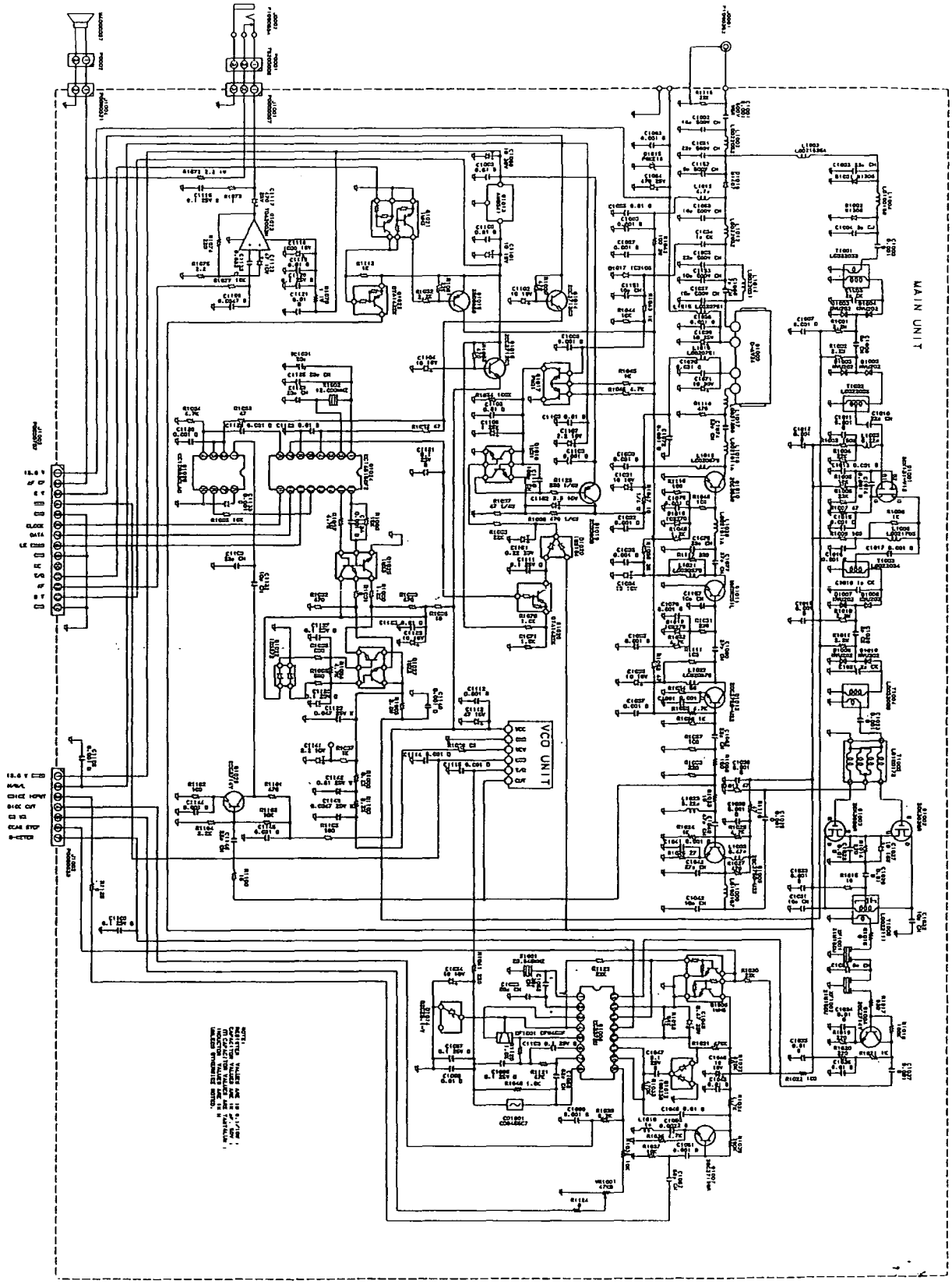

Fig. 4.8 Main Unit Circuit Diagram Of YAESU FT-2400H Transceiver

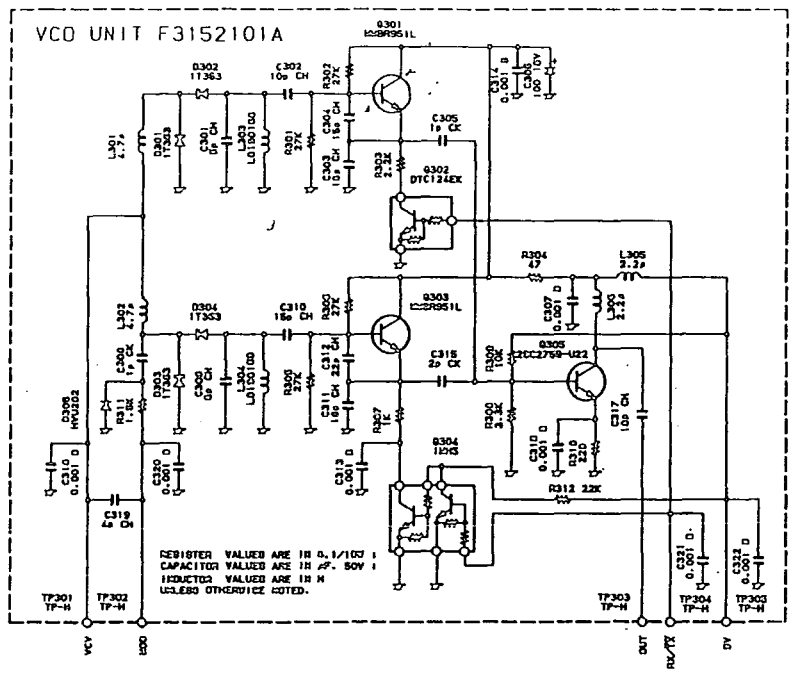

Fig. 4.9 YAESU FT-2400H VCO Unit Circuit Diagram

References

1. Naylor J,1991. *An Introduction To Packet Satellites.* OSCAR NEWS, The Official J. Of AMSAT-UK For All Users Of Oscar Satellites, No. 90 pp 12-17, August.
2. Dave, 1991. *UOSAT-3 Operations Update.* OSCAR NEWS, The Official J. Of AMSAT-UK For All Users Of Oscar Satellites, No. 88 pp 17-19, April.
3. Ward J, 1991. *UOSAT-F Pre-Launch Information.* OSCAR NEWS, The Official J. Of AMSAT-UK For All Users Of Oscar Satellites, No. 90 pp 8-10, August.
4. Bailey T, 1983. *Circular Polarisation Reaches The Parts Other Polarisation Cannot Reach.* Ham Radio Today, pp 31-33, March.
5. PacComm Packet Radio System Inc. 1991. *Operating Instructions, TINY-2 And MICROPOWER Packet Controller.* PacComm Inc.
6. PacComm Packet Radio System Inc. 1991. *Operating Instructions, MC-NB96 And EM-NB96 Narrowband 9600 Baud Packet Modem.* PacComm Inc.
7. King G. J, 1970. *The Practical Aerial Handbook.* Newnes-Butterworths, London, 2ED.
8. Stark H, Tuteur F. B, Anderson J. B, 1988. *Modern Electrical Communications (Analogue, Digital, And Optical Systems).* Prentice-Hall, Englewood Cliffs, New Jersey, 2ED.
9. PB and PG Users Documents - 0.003. 1992.
10. Horzepa S, 1989. *Your Gateway To Packet Radio.* ARRL, Newington, CT USA 06111, 2ED.
11. Hutchison L. C, et al. (editors), 1985. *The ARRL Handbook For The Radio Amateur.* ARRL, Newington, CT USA 06111, 2ED.
12. Davidoff M, 1990. *The Satellite Experimenter's Handbook.* ARRL, Newington, CT USA 06111, 2ED.
13. Bleazard G. B, 1982. *Handbook Of Data Communications.* NCC Publication.
14. Sherman K, 1981. *Data Communications - Users Guide.* Reston Publishing Company, Inc.

15. Wiesner L, 1981. *Telegraph And Data Transmission Over Short-wave Radio Link s - Fundamental Principals And Networks.*
SIEMEN Aktiengsellschaft, Heyden & Son Ltd, 2ED, 1981.
16. Curtis , 1991. 9600 Baud Packet
17. Miller J. R G3RUH, 1988. *9600 Baud Packet Radio Modem Design.*
Proc. of 7th ARRL Computer Networking Conference, pp 135-140,October.
18. Miller G. M, 1988. *Modern Electronic Communication.*
Prentice-Hall International, Inc. 3ED.
19. Reisert J. M, 1987. *How to Design Yagi Antenna.*
Ham Radio, pp 22-31, August.
20. Sato G, 1991. *A Secret Story About The Yagi Antenna.*
IEEE Antenna And Propagation Society Magazine, Vol.3.3 No.3 pp 7-18,June.
21. Cheng C.K, 1991. *Gain Optimisation For Yagi-Uda Arrays.*
IEEE Antenna And Propagation Society Magazine,Vol.3.3 No.3 pp 42-45, June.
22. James R. J, 1990. *What's New In Antenna.*
IEEE Antenna And Propagation Society Magazine, pp 6-18, February.
23. Crisler M, 1991. *The Pacsat Beginner's Guide.*
AMSAT-UK Publication.
24. Ward J,1988. The UOSAT-D Packet Communication Experiment.
ARRL 7th Computer Networking Conf. Proc. pp 186-193, Columbia, Maryland.

Chapter 5

SATELLITE TRACKING

5.1 Introduction

Amateur satellites launched so far, are polar orbiting satellites, hence, they are only periodically accessible by the ground-station. In order to communicate through them it is important to know when they will be in range, where to point the directional antenna and what region of the earth surface is accessible at that particular time, through the process of satellite tracking. This chapter describes the methods and a system used in tracking LEO satellite such as OSCAR 22.

5.2 Tracking Overview

LEO satellite moves rapidly across the sky with a typical period ranging from 90 - 120 minutes. A satellite will be in range of a ground-station for a limited amount of time, commonly referred to as a satellite pass. The duration of the pass depends on (i) the orbital period of the satellite, and (ii) the distance it will be at the point of the closest approach to the ground-station. The longest duration will occur during the overhead pass [6]. For instance, OSCAR 22 will generally be in range for less than 20 minutes every time it passes a station with 4 to 6 satellite passes a day. Therefore, to access such satellites efficiently it is important to know when they will be in range and where to aim the directional antenna.

The process of knowing the position of the satellite in space at a particular time, and co-ordinate relative to the ground-station, is what is known by scientists as satellite tracking. It generally involves geometrical calculations of orbital elements to give a correct description of a satellite position in

space at a particular time. However, for radio amateurs tracking refers to the practical concern of when the satellite will be in range (AOS and LOS times), where to aim directional antenna (Azimuth and Elevation), and the accessible region of the earth within the spacecraft footprint at any particular time. In amateur radio, calculated orbital elements usually supplied from AMSAT organisations are used in different tracking methods developed by radio amateurs. In other words, radio amateurs can track LEO satellites without needing to know the basic physics of satellite motions or how a satellite moves in space.

5.3 Geometry Of Orbiting Satellites

The motion of the planets around the sun has been a subject for many years, Kepler's laws and Newton's laws of motion are typical examples of such early studies. As a result, it has been possible to use spherical geometry equations to describe planet's orbits. The laws that have been derived from these observations are equally applicable to amateur satellites.

To determine the path of the satellite in space, assumptions based on the Keplerian model are required. Assuming that (i) A satellite orbits in an ellipse of constant shape and orientation (ii) The earth and a satellite are spherically symmetry and forms a two body system whose only force acting upon them is the attraction force, the path of the satellite can be calculated using spherical geometry. These assumptions simplify calculation of the orbital equation, although in reality corrections to the solutions obtained may be required. Consider the first rectangular co-ordinate system below used to describe the earth and the satellite in Fig. 5.2.

Assuming that this holds in space, and the earth rotate in the z-axis. The gravitational force $\mathbb{F}$ on the satellite is given by

$$\mathbb{F} = \frac{-GM_E m_s}{u^2}\hat{\mathbb{u}} \quad \text{..... (5.1)}$$

where M_E = mass of the earth,

G = universal gravitational constant,
$\hat{u}$ = the unit vector in the u direction,
$\mu = GM_E$ the Kepler's constant.

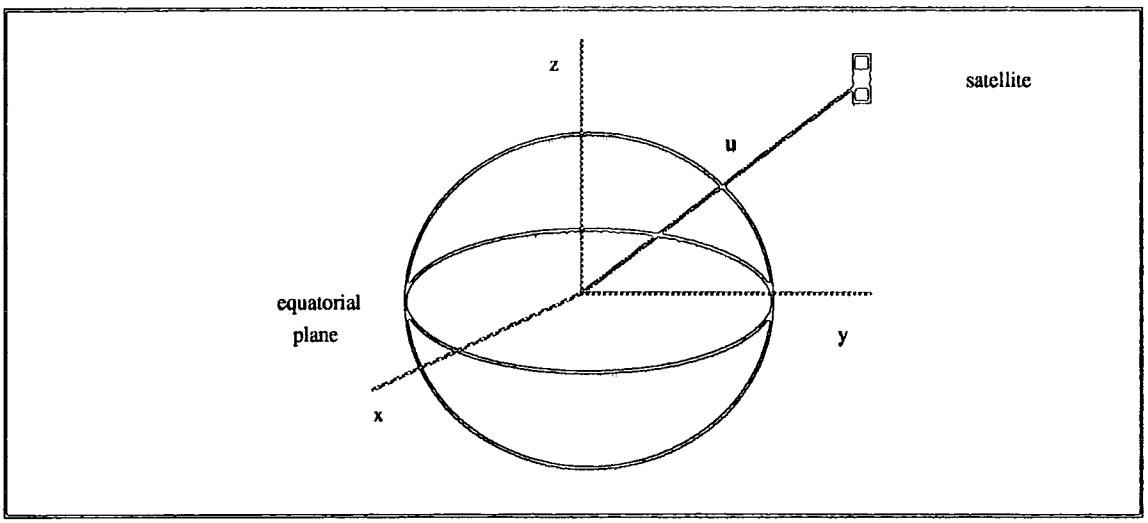

Fig. 5.2: The first co-ordinate system used to describe the earth and a satellite. The vector $\mathbb{u}$ locates the moving satellites with respect to the centre of the earth.

From Newton's second law of motion

$$\mathbb{F} = m_s \frac{\partial^2 u}{\partial t^2} \hat{u} \quad \text{..... (5.2)}$$

If we equate equations, (5.1) and (5.2), and define $\mathbb{u} = u\,\hat{u}$, we will have the following equation denoting the satellite motion in space.

$$\frac{1}{u}\frac{\partial^2}{\partial t^2}\mathbb{u} + \frac{\mu}{u^3}\mathbb{u} = 0 \quad \text{..... (5.3)}$$

Eqn.(5.3) is a second order vector linear differential equation whose solution will require six undetermined constants called "Keplerian" elements (after Johann Kepler [1571-1630]). These elements define an ellipse, orient it about the Earth, and place the satellite on the ellipse at a particular time. They consists of three position co-ordinates and three

velocity components, usually expressed in an inertial Cartesian co-ordinate system [1].

5.3.1 Orbital Elements

A set of commonly used orbital elements in satellite communications are; eccentricity (e), semi-major axis (a) or mean motion, mean anomaly (M) or apogee time (t_p), right ascension of ascending node (Ω), inclination (θ) and argument of perigee (ω) [4].

5.3.1.1 Eccentricity

The eccentricity defines the degree of "loopsidedness" of the orbit. If eccentricity e = 0, the orbit would be a perfect circle. If $0 < e < 1$, the orbit is an ellipse, e = 1 it is a parabola, and $e > 1$ it is a hyperbola.

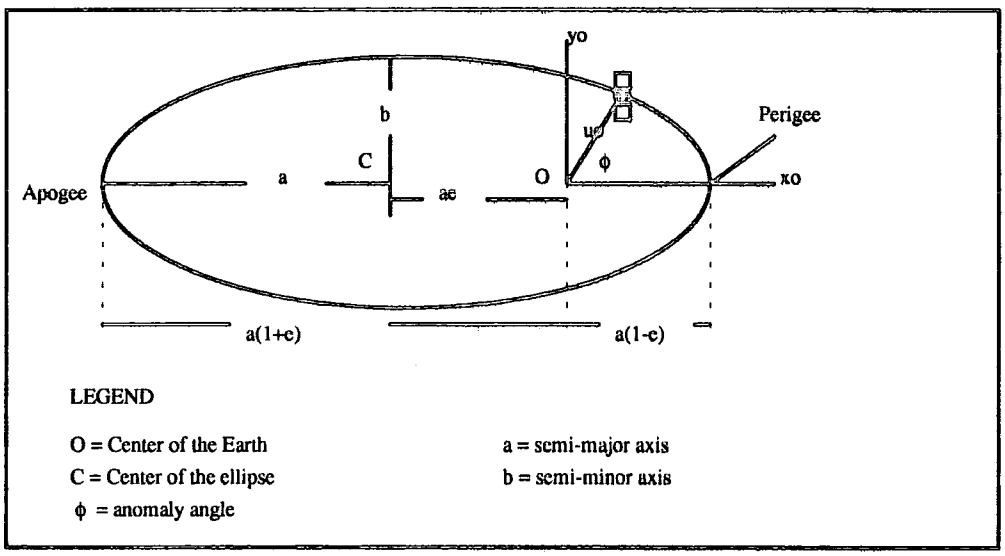

Fig. 5.3, The orbit as it appears in the orbital plane.

5.3.1.2 Mean Anomaly

Mean anomaly is arc length in radians, described by the satellite since the perigee passage if it were moving at the mean angular velocity $\eta = 2\pi / T$,

where T is an orbital period. Specifically, it is a measure of time since perigee expressed as an angular quantity from 0 - 360 degree per orbital period. It must be noted that satellites in elliptical orbit does not move at constant rate, hence, the angle represented by the mean anomaly does not correspond to any measurable, physical angle. However, a true anomaly (E) can be computed with the help of fig. 5.4, which is an angle as seen from the centre of the earth measured in the direction of satellite motion, from the perigee point and the satellite's current position in the arbitrary circumscribed circle. Mean anomaly can be calculated using the derived equation below.

$$M = (t - t_p) = E - e\sin E \quad \text{..... (5.4)}$$

where M = Mean anomaly,

E = Eccentric anomaly = arc $\cos((a - u_o e)/ae)$,

t_p = time at perigee,

t = arbitrary time.

5.3.1.3 Epoch Time

Strictly speaking, this is not an orbital element. It is a seventh element found in many supplied Kepler element sets to indicate an instant time at which other numbers are valid. This number can be chosen arbitrarily by the individual when generating the element set, but usually chosen near the middle of the observation times used to generate the set.

5.3.1.4 Mean Motion

This is the number of complete orbits the satellite makes in a day. It is simply the reciprocal of the orbital period. Once the mean motion is known, a quantity called semi-major axis can be calculated.

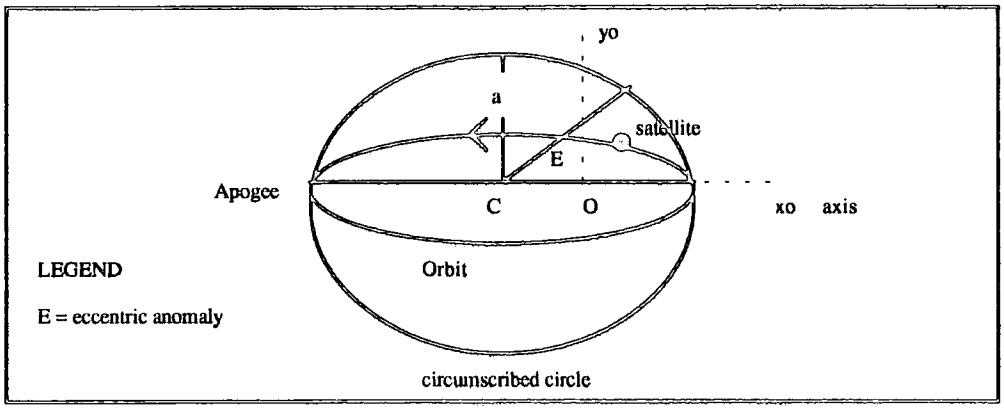

Fig. 5.4 Circumscribed circle where true or eccentric anomaly E is calculated [1].

Provided that, Mean Motion, Mean Anomaly and Eccentricity are known, the co-ordinates (u_o, ϕ_o) and (x_o, y_o) of the spacecraft in the orbital plane can be specified, hence describing the shape and size of the orbit. The following elements specify how the satellite can be located from a point on the rotating surface of the earth.

5.3.1.5 Argument Of Perigee

Argument of perigee is the angle ω measured in the orbit plane in the direction of motion of the satellite, between the equatorial plane and the perigee point (Fig. 5.5).

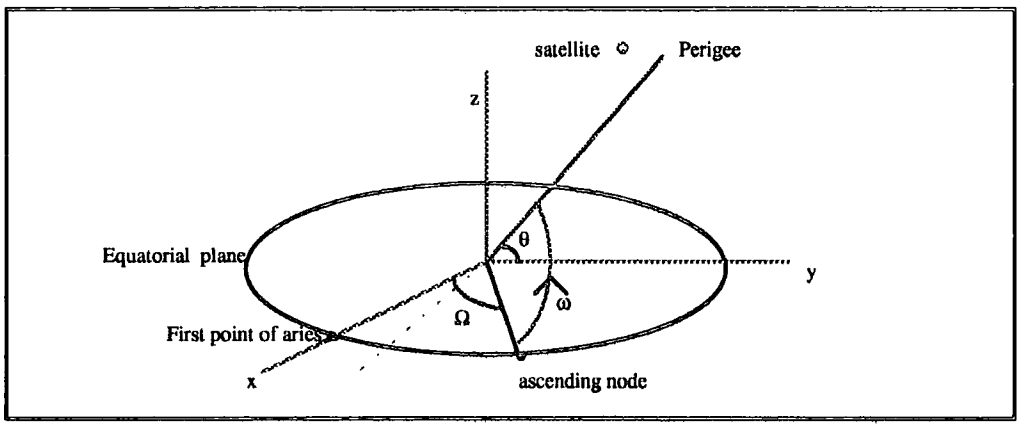

Fig. 5.5 The geocentric co-ordinate system showing the inclination (θ), argument of perigee (ω) and Right ascension of ascending node (Ω) angles.

5.3.1.6 Inclination

Inclination (θ) is the angle between the orbit plane and the earth's equatorial plane (Fig. 5.5). An inclination of 90 degrees indicates that the satellites passes over both poles. An inclination of near 90 degrees form a polar orbit, which is a typical characteristic of many polar orbiting satellites.

5.3.1.7 Right Ascension Of Ascending Node

Right ascension of ascending node (RAAN) is the angle Ω measured along the equator, between the ascending node and the first point of Aries (Fig. 5.5). The first point of Aries is defined as the point at which the sun crosses the equator into the Northern Hemisphere.

Calculations of these elements usually is not a requirement in many amateur tracking methods. They are supplied by different organisations such as AMSAT-UK, NASA, and many others of similar nature, only required in tracking software or used to generate orbital calendar which provides essential information for graphical tracking methods.

5.3.2 Azimuth And Elevation

The tracking system at the earth's ground-station requires antenna pointing data in the direction of the satellite once in range. The co-ordinates to which an earth station antenna must be pointed to communicate with a satellite are called Azimuth and Elevation. Azimuth is measured eastward from geographic north to the projection of the satellite on a (locally) horizontal plane at the earth station. Elevation is the angle measured upward from the horizontal plane to the satellite path. The following derived equations can be used to calculate the antenna pointing data required by the ground-station tracking system and are usually implemented in many tracking software [2]. A proof of these equations are not going to

be explained, indeed it is not necessary to understand them fully to make constructive use of them. If we define,

L_s = subsatellite latitude in degree,
G_s = subsatellite longitude in degree,
L_e = equator crossing longitude in degree,
L_o = user's latitude in degree,
G_o = user's longitude in degree,
t = time after S-N equator crossing (ascending node),
T = period,
θ = inclination,
A = azimuth,
E = elevation,
R = radius of the earth,
h = height of satellite.

The subsatellite latitude at time t after crossing the equator is

$$L_s = \arcsin\left[\sin\left(\frac{360t}{T}\right)\sin\theta\right] \quad \text{.....(5.5)}$$

The corresponding longitude is

$$G_s = \arccos\left[\frac{\cos(\frac{360t}{T})}{\cos L_s}\right] + \frac{t}{4} + L_e \quad \text{..... (5.6)}$$

In order to calculate the position in the user's sky the angle at the earth's centre between the subsatellite point and the user must be calculated.

$$\theta = \arccos\left[\sin L_o \sin L_s + \cos L_s \cos L_o \cos(G_s - G_o)\right] \quad \text{..... ..... (5.7)}$$

The azimuth from user's location is now

$$A = \cos\left(\frac{(\sin L_s - \sin L_o \cos\theta)}{\sin\theta \cos L_o}\right) \quad \text{..... (5.8)}$$

and the elevation

$$E = \arctan\left(\frac{\cos\theta - \frac{R}{R+h}}{\sin\theta}\right) \quad \text{..... (5.9)}$$

In this way more accurate figures may be obtained for the time between equator crossings and AOS. The direct distance between user and satellite may be calculated using the following equation.

$$d = \sqrt{(R+h)^2 + R^2 - 2(R+h)R\cos\theta} \quad \text{..... (5.10)}$$

These are essentially repeat calculations as the satellite moves, so they are most easily handled by a computer. The described equations above, are fundamental for all tracking methods which will be described below.

5.4 Tracking Methods

There are two tracking methods in application today; the traditional graphical method and the modern computer method. Currently, computer methods are popular among amateurs, and are rapidly replacing the traditional graphic methods. This is due to the advanced technology in computers which has made PCs affordable by many amateurs. Although traditional methods are replaced rapidly, they are very useful in understanding what is displayed by the computer graphics.

5.4.1 Graphical Methods

Different graphical methods are used to track satellites depending on whether the satellite is in (i) low altitude circular orbit, (ii) elliptical orbit and (iii) geostationary orbit [3]. Different graphical tracking methods have been developed by radio amateurs since the inception of amateur satellites. One of the popular graphical methods is the OSCALATOR, a tracking device which is suited for tracking LEO satellites. In order to use this

method an orbital calendar, usually supplied by AMSAT organisations, is required. The orbital calendar supplies information about the number of equatorial crossing times for different passes of different satellites, to enable the user to calculate the AOS and LOS times, and antenna pointing data for the ground-station [5]. This method has only been used once at this station while trying to verify the accuracy of the antenna pointing data from the computer program. More information on the use of this method can be found in the Martin Davidoff's book "Satellite Experimenter's Handbook".

5.4.2 Computer Methods

Tracking satellites by computer methods is by far the most convenient way. There are many simple and more complex, tracking software on the market today, written by different people for different computer families. Tracking software provides data about the time the satellite will be in range (AOS and LOS time), the direction to point beam antenna, Doppler shift corrections and many other parameters depending on the complexity of the program. The available tracking software at Durham University station is the commercial DK1TB SAT-PC satellite tracking program for IBM PCs and clones which can be used in conjunction with the RICOFUNK/AMSAT IF-100 rotor interface device for rotating the antenna by computer [7].

Any tracking software needs Keplerian elements to update the correct orbital determination of satellites involved. These elements are critical in tracking satellites as described before and they should be updated after every four months for LEO satellites such as OSCAR 22, because the orbital plane shifts with time. AMSAT-UK distributes these elements in every OSCAR NEWS Journal or they can be downloaded from orbiting PBBS on board pacsats (Appendix 1).

5.5 Tracking System

Satellite tracking systems differ depending on many factors, one of them is purchasing power of the user. Someone using graphical methods will have different tracking systems compared to those achieved through computer methods. The most important fundamental minimum parts for any tracking system, regardless of its tracking method orientation, is the rotator and rotator controller. These enable the user to point antenna in the right direction without difficulty, although such minimum requirements demand the physical presence of the operator.

However, for any serious user of OSCAR satellites, the best tracking systems although expensive, are computer controlled systems, due to the fact that the orbit mechanics of LEO satellites together with other factors, make it practically impossible for an individual operator to efficiently track the satellite manually. For instance, due to Doppler effects the tuned receiving frequency needs to be adjusted to compensate for Doppler shift, at the same time an operator should rotate the antenna in the direction of the satellites. Even more problematic is a situation when the transmitter simultaneously needs attendance. All these demand a physical presence of an operator during satellite passes. Hence, the tasks frequently become too demanding for a single operator during a pass.

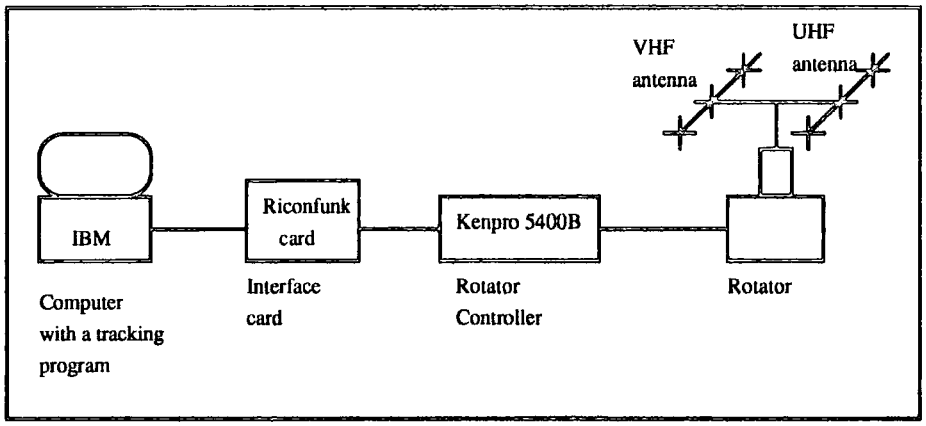

Fig. 5.3 Computer Controlled Tracking System Block Diagrams.

A computer controlled tracking system basically requires a rotator, rotator control, and an interface card between the rotator controller and a computer. Most of these components of the tracking system are commercially available. The block diagrams of fig. 5.3 summarises the semi-automatic computer controlled tracking system set at Durham University ground-station.

The two crossed Yagi antenna for VHF and UHF band plan at this ground-station are mounted on the same beam parallel to each other. This beam is mounted on the antenna rotator system which is used to rotate the two antennas when tracking the satellites. The VHF antenna is a six elements crossed Yagi antenna. The UHF antenna is an 18 element crossed Yagi antenna. In order to re-orient the beam as the satellite moves across the sky, the azimuth and elevation of the two antenna are operated by the KENPRO 5400 elevation-azimuth dual controller to rotate the antenna rotator. This controller can also receive antenna pointing data from the tracking program by connecting to it an interfacing device called RICOFUNK/AMSAT IF-100. In other words, the re-orientation of the beam can be controlled automatically by the computer program.

5.6 Accurate Prediction Of Satellite Passes

The tracking software gives a lot of information about any selected satellite. How accurate the information is requires verification. A simple experiment was carried out by listening to many amateur satellite downlink signals, (SSB, FM and CW signals) at the predicted time and position. After updating the Keplerian elements in the program, carrier signals for different satellites were detected as predicted by the program. A packet sound, similar to terrestrial packet sounds, could be heard for satellites using 1200 AFSK baud system such as OSCAR 11. A predicted satellite pass calculated by the Graphical methods, gave minor differences in AOS time and antenna pointing data compared to software calculated data, although produced the same results. Bulletin news have been downloaded from

OSCAR 23 (appendix 1). This is sufficient to confirm that the detected downlink signals are coming from the expected satellites and not otherwise.

Although supplied orbital element sets for LEO satellite can last up to four months, it has been found that old and fast decaying satellites such as OSCAR 11, require fresh orbital elements to give correct descriptions of the satellite in space. Older orbital elements result in inaccurate prediction of satellite passes, a serious problem in automatic tracking. Observations in this project using OSCAR 11 show that orbital elements, at most two months old, can be used without difficulty in tracking fast decaying satellites.

5.7 Summary

Accuracy of tracking a satellite is important if we have to communicate efficiently through these satellites. In computer methods, Keplerian elements must be constantly updated, and the time on the computer running the tracking software as accurately as possible, down to seconds of accuracy. Fast decaying satellites require fresh orbital elements, good performance can be achieved if they are changed after every one month. Both graphical and computer methods give an acceptable approximation of the actual position of the satellite in space. However, though relatively expensive, computer methods are more convenient in tracking satellites. The next chapter presents the process of downloding and uploading from and to the satellite and discusses the performance of the process with respect to still-continuous tone image transmission.

References

1. Pratt T, and Bosnian C. W, 1986. *Satellite Communication.* John Wiley & Sons Ltd.
2. Feedback Weather Satellite Receiver WSR 53 Vol. I, 1985. *Operation Manual.* Feedback Instrument Ltd.
3. Davidoff M, 1990. *The Satellite Experimenter's Handbook.* ARRL Publisher, Newington CT USA 06111, 2ED.
4. Wallach J, 1991. *A Primer On Satellite Orbital Keplerian Element Formats.* OSCAR NEWS, The Official J. Of AMSAT-UK For All Users Of Oscar Satellites, No. 89 pp 32-37, June.
5. Spencer S. J, 1985. *The Sheffield Project Satellites OSCAR 9-Book 1.* AMSAT-UK Publisher.
6. Crisler M, 1991. *The PACSAT Beginner's Guide.* AMSAT-UK Publisher.
7. DK1TB SAT-PC Satellite Tracking Program, *User's Manual.*

Chapter 6

DATA DOWNLOADING AND UPLOADING

6.1 Introduction

Any data which can be stored in a form of a computer file can be used in pacsat store-and-forward e-mail communications. Data downloading and uploading is done in a similar way as in many computer client-server systems. The server on board the satellite will know nothing about the contents of the files nor will it take any active role in forwarding them, rather it will know how to receive and transmit any file in the specified file header standard format described before. This chapter pressent the process of downloading and uploading of messages and image data, and outlines the results obtained, by discussing the performance of the two processes with respect to image transmission.

6.2 Message Downloading

Message downloading from OSCAR 22 by the ground-station is easy with the previously explained satellite message broadcasting system. The downloading of files from the PACSAT server on board the spacecraft is done using the PB.EXE program. This program implements the PACSAT Broadcast protocol. Before running this program, the computer must be configured using PB.CFG file to suite the individual need of the ground-station [4].

Message transmission from OSCAR 22 to ground stations is a point-to-multipoint transmission. The Broadcast protocol transmits UI frames which allows many stations at the satellite footprint to receive data from the spacecraft without being connected to the server on board. On running the

program in its auto mode, it will rest (waiting state) when the satellite is out of range. Once the satellite is in range the program waits for the first UI frame before starting automatic communication with the satellite. Assuming no file has been marked for downloading by the file status markers, the client (G1YNA, our ground-station callsign) program will initiate communication (for station with transmitter) by sending a directory update request. In response the PACSAT server will transmit all directory and file header entries in the server which can be seen in the active window of the PB menu which report the activities of PB program to the user. If GOK8 is using the server, it will put your request in a queue, and transmit a beacon 'PB:\GOK8\G1YNA' which will be displayed in the downlink window which shows the activity of the satellite server.

6.2.1 Selecting File For Downloading

After updating the terminal directory, a directory view service can be used to view the message headers of all messages in the PACSAT server which have been heard by the ground-station. By using this service a file of interest can be selected for downloading. The selection of a file of interest is done by marking a file with a letter corresponding to any of the status markers; A automatic, P priority, G grab, and N never [4]. If the file is marked A, the standard automatic, the message will be completely downloaded and holes will be filled automatically, without the intervention by the PB program. Once the message has been completely filled, it will go to the message directory MSG, usually a separate directory within the PB program receding directory. A completely downloaded message in this directory will have an extension xxx.DL. It is at this stage that the completely filled message is striped off its file headers by the PHS service which at the moment is part of the PB program. The downloaded file in this message directory has some meaningless characters, associated with stripped headers, which can be removed by further processing of the message file using any word processor. However, this meaningless characters do not interfere with the actual message which is written in the

127 ASCII standard characters. An example of the downloaded file from OSCAR 23 can be found in appendix 2.

6.2.2 Downloading Performance

The observed speed of downloading a message of interest from the PACSAT server depends on many factors, notably (1) the number of users within the satellite footprint, (2) the field strength of the arriving signals, which is subject to attenuation and depolarisation along its path, and (3) on-board computer crashing which necessitates closure of the service for reloading of software by the commanding station. Following is a discussion of the implications of each of these factors.

6.2.2.1 Number Of User's

The number of ground-stations within the satellite's footprints interested in the message to be downloaded determines how fast the downloading process of a such message will be. Many microsatellites have a limited number of simultaneous users, only a small percentage of users within the footprint, usually with powerful transmitters, can communicate with the satellite's server. Suppose many users within the satellite footprint are interested in the same file, all ground-stations will be able to receive the message regardless of the initiator of message downloading, further, for stations with transmitters, every 'hole fill' requested by any station will be received by all stations so long as the packet (hole) transmitted by the satellite has not been received before by the concerned ground-station. In this way no matter how long the queue is, chances of getting more data in a single pass is guaranteed, hence fast reception of the message of interest. Downloading of such files will take a short time depending on the size of the file. But if only one user is interested in such a file, the probability of the station's hole request to be accepted by the server decreases with the increases of number of users within the footprint. With limited time for a single pass, there is a possibility of getting access to the PACSAT server

when the station is almost out of range by that time. This is more significant in areas with high usage such as UK and Europe in general. It must be noted that, most of the explanations above are for ground-stations with transmitters, for receiving only ground-stations, they have no choice but to receive whatever has been broadcast. As a result, the message downloading speed of such stations will always depend on other stations within the satellite footprint.

Therefore, the speed of downloading is a function of the file size, the number of users within the satellite footprint, and the popularity of the file being downloaded as explained above. A popular file size of 5,000 bytes may take 10 passes to completely download, but a similar unpopular file may require up to 30 passes or more.

6.2.2.2 Downlink Signal Strength

The strength of the downlink signals arriving from OSCAR 22 and OSCAR 23 is continually changing. Although this is expected due to fading, atmospherics, interferences from other radio transmitters, and other effects, these variations of the signal strength causes downlink of data at the ground-station to be non uniform for similar satellite passes. My observartion at Durham University station shows that it is not guarantee that whenever the satellite is in range, the ground-station will receive data efficiently all the time for all passes. Some days have produced better results than others, making general data reception inefficient. This may be due to some of the following reasons.

Amateur satellites are experimental satellites (as explained in chapter two), their transmitting powers are limited depending on their designs. For instance, OSCAR 22 has been designed to transmit continuously with 5 W power, in order to use the batteries on-board the satellites economically, hence extending their life span throughout the satellite lifetime. Conversely, in order to transmit to the spacecraft, ground-station transmitters capable of producing 25 W transmitting power are widely used. Although the 5 W

transmitting power has been found to work well from a designer's point of view, the downlink signal is vulnerable to attenuation and depolarisation caused by different effects in critical conditions, particularly for users with simple antenna system.

In the UK, radio amateurs are secondary users of the UHF frequency band plan. The primary user of this band plan is the Ministry Of Defence, which uses it for 70 cm military radar systems. Interference from their transmission is likely to cause poor reception to some ground-stations within UK.

6.2.2.3 On-Board Computer Crashing

Amateur satellites are experimental satellites which carry other experimental payloads, in addition to the digital store-and-forward communication system payload. Different experiments are always carried out by the commanding stations. Unfortunately, some of these experiments could go wrong causing the on-board computer system to crash. Also, some users may unknowingly upload files with computer viruses causing the computer system to mal-function or crash. As it turns out, these problems happens, forcing the commanding stations to close the service and reload the software. Any file which was not put in broadcast circulation by the commanding station will be lost unless the file is uploaded again.

6.3 Message Uploading

Message uploading to the PACSAT server is done using the PG program. This program implements the FTL0 protocol to establish a link between the client and a pacsat server. Similarly, a PG program must be configured to suite the station before using it for transmission [4]. Uploading messages to the PACSAT server is difficult compared to message downloading. This is due to the fact that the transmission between the two stations is a point-to-point communication, i.e. only one client is allowed to communicate with a

server at one time (virtue-circuit mode of AX.25 protocol). Once a link has been established, the file transfer process will begin and will be terminated only when the file has been completely uploaded, unless the communication link is interrupted.

6.3.1 Preparing A File For Uploading

In order to upload a file to the PACSAT server, the file must be prepared in a standard format acceptable by the client and server programs. This implies including headers in a file to be uploaded which is done by the utility program PFHADD program. No file will be uploaded by the PG program unless the file has been prepared by this utility program. The PFHADD program implements a flexible encoding method for PACSAT file headers [3]. These headers are standard headers which can be found at the beginning of every file downloaded from the PACSAT server (or any fully compliant FTL0 server) to ensure that files can be properly identified and processed in the PACSAT server (destination, uploader, filename, original filename, etc.). This allows clients to determine the status of the file on the server. More specifically, the originator client may wish to find out if the file has been downloaded by its intended destination, and a Gateways client may wish to find out if a message still needs to be downloaded for relay to a particular destination.

6.3.2 Logging Into The Satellite Server

The prepared file by the PFHADD program will have an extension xxx.OUT symbolising that the file is ready for uploading. On running the PG program, it will be searching for files with this extension once the upload command has been selected from the menu, and then continue to another stage. For OSCAR 22 and similar PACSATs operated by the University of Surrey, the program will go into WAITING state (waiting for free user position on the satellite), where it will be waiting for a BBSTAT current status beacon message which is transmitted every five seconds. If

the server is free the message reads: "UOSAT5-12>BBSTAT: Open 2: G1YNA". Once the satellite is in range and the beacon message has been heard, the program will automatically try to establish a link to the satellite by issuing a connect request. In doing so it may go in any of the following states; WAITING, BACKOFF, REQUEST and LINKED states depending on the status of the server [4]. Once the connection has been established, the PG indicator will show 'LINKED', and the communication function will proceed. Otherwise, the PG will return to the 'WAITING' state. After the file has been uploaded the PG program will disconnect.

A message file has been uploaded to the satellite and then downloaded after one day stay on-board both OSCAR 22 and OSCAR 23 satellite servers (appendix 3). Once linked the server will transmit a message "upload accepted" at the same time assigning the file 'xxx.out' ('g6.out' in this case), a file number (F7AB) which will be used to broadcast the message once it is completely uploaded. If, for various reasons, the process is terminated before a file is completely uploaded, the xxx.out changes status to xxx.pul indicating that the file is partially uploaded. In the next pass, any file with extension xxx.pul (F7AB.PUL) will be given first priority for uploading by the PG program. Once the file is completely uploaded, the status of the file changes to xxx.ul (F7AB.UL) indicating that the file has been completely uploaded. After a few passes, the file will be broadcast by the satellite and can be downloaded using the PB program as described earlier.

6.3.3 Uploading Performance

The uploading performance at the Durham University ground-station is affected by the hardware design limitations on radio transmitter used. Large message files cannot be uploaded using this transmitter with the modifications done. The YAESU FT 2400H transceiver is not one of the recommended rig for 9600 baud data transmission operation [5]. On top of having a narrow bandwidth (10 kHz), it has a PLL circuit which tends to stop modulation of a carrier during transmission as explained in chapter 4.

However, other limiting factors affect the uploading process, some of which are similar to those explained in the downloading performance. Since the communication between the ground terminal and the satellite server is point-to-point, i.e. one user is allowed to communicate to the satellite at anyone time. The probability of logging into the satellite decreases as the number of users within the satellite footprint increases within the available time of the satellite pass. If the downlink is poor, there will be an unstable link established which could easily be terminated because of lost contact. One of the more serious problems is the use of the service by gateways stations (sometimes referred to as satgates) as is in OSCAR 22, which use the service to route many files accumulated from terrestrial packet radio stations. Once such a station has logged into the satellite server, no other user will be allowed to use the service until all the files have been uploaded. By the time the gateway is disconnected, assuming it has a few files, the pass will be almost over. Similarly, during the downloading process these files are broadcast by the spacecraft, causing receiving stations to receive a lot of unwanted message directories. This is a serious limitation to individual users of the service. As a result, informal agreement between users has been reached where gateways use OSCAR 22 only while OSCAR 23 is used by individual users [6].

6.4 CCD Images Downloading

The development of high-density semiconductor Charge-Coupled Device optical detectors, have presented a new opportunity for remote sensing using microsatellites. This new development has led to the development of the Earth Imaging System (EIS) on board OSCAR 22 and OSCAR 23 microsatellites. OSCAR 22 is the first microsatellite to have an EIS on board capable of taking images routinely of the Earth surface. The satellite uses the EIS based on CCD camera on board to take images of the earth. The earth images are stored by the CCD camera as bitmaps. The size of images from this system is a 576 x 567 pixels with 256 grey levels, which, when put into bulletin board on-board the satellite, will have the size of approximately 350,000 bytes. The EIS on board OSCAR 23 is the advanced

EIS following the success made from the previous EIS. Two images, wide field angle with a ground resolution of 850 m, and narrow field angle with a ground resolution of approximately 300 - 400 m, can be taken from this advanced EIS whereas only a wide field angle was considered in OSCAR 22 [2].

The downloading of image files from the satellites is done in the same process as in messages downloading explained above. Due to the nature of operation of the satellite and its broadcasting protocol, together with other reasons as explained before, it takes a number of days to download a big file. Examples of CCD images from OSCAR 22 wide field angle and OSCAR 23 narrow field angle cameras are shown in fig. 6.3.1 and fig.6.3.2. These images are of medium resolution, and meteorological quality scale comparable to the NOAA and METEOSAT satellites [2]. The process of downloading CCD images is very frustrating in areas with many active users and gateways (for OSCAR 22), particularly when most of them are not interested in downloading the image. As a result, downloading CCD images in the present situation is inefficient.

The CCD EIS was put on OSCAR 22 as an experimental payload, but the success of this experiment exceeded the expectations of the engineers at Surrey University. Image gathered reveal a surprising amount of detail. As a result of this, they were not prepared to handle these images to other users in an efficient manner compared to other meteorological satellites. For instance, up to the time of writing of this report there is no software certified to display the downloaded images, rather it is up to an individual to process these data anyhow and anywhere. Further, sometimes it is difficult to tell which part of the earth the picture exactly represents unless the commanding station gives such information. Due to the limited amount of storage memory in the spacecraft, not many images can be put into circulation at the same time, for instance OSCAR 22 is capable of storing a maximum of 800 messages which are shared by other users.

However, there is still a lot of promising research going on in this area to improve the performance of the EIS. Some of the achievements can be seen

in the OSCAR 23 EIS ground resolution power using narrow field CCD camera, an improved version of OSCAR 22 EIS, providing a ground

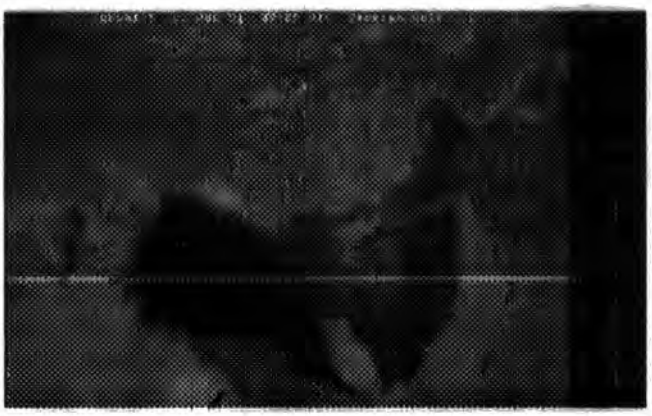

Fig. 6.3.1 OSCAR 22 Wide Field Angle CCD Image.

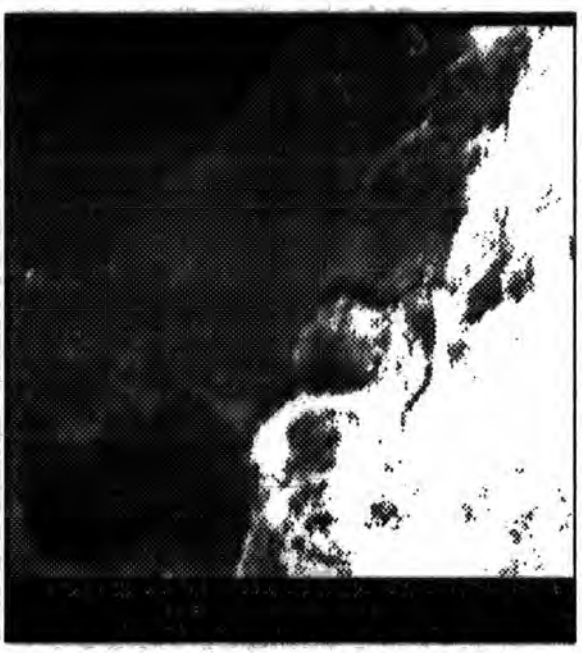

Fig. 6.3.2 OSCAR 23 Narrow Field Angle CCD Image.

Fig. 6.3.3 A decompressed image uploaded then downloaded from the pacsat serve

resolution of approximately 300 - 400 m with 650 nm filter. More interestingly, the advanced Transputer Image Processing Experiment (TIPE) on board OSCAR 23 is investigating the implementation of exposure assessment routines optimising the dynamic range of the images, basic image acquisition to remove noise and striping effects, and merits of different image compression and geometric distortion compensation routines. Images will be converted from their row format to the GIF standard, so that existing viewers can be used to display the images [1]. An advanced multi-spectral EIS, similar to those in meteorological satellites, for use with the UOSAT microsatellite platform is under development and will use three 1000 x 1000 pixel CCD arrays with ground resolution of 100 m [2].

6.5 Image Data Transmission

So far we have seen how the microcomputers are increasingly becoming useful in data communications. Digital image technology has also, contributed to transforming microcomputers into visual communication tools. Recent developments in computer related technologies, in the area of 32-bit microprocessors, RAM 4 megabytes, and advanced hard disks have made possible the application development of microcomputers for digital processing of real-world pictures. Recently, the application of microcomputers for visual communications has become popular in many areas such as multi-media information technology. Therefore, visual communications through Health-Net is likely to be an example of this transformation in the near future.

Uploading and downloading of images is done in a similar manner as in message uploading and downloading explained above. The transmission of image data which are usually associated with large files, suffer similar limitations as explained in message downloading and uploading performances. For the purpose of demonstration, a small image file has been uploaded to the satellite and downloaded in a later pass (fig. 6.3.3). The downloaded file which need further processing to remove the file

header, has been processed using the Microsoft WRITE accessory program to remove this header and can be displayed using any standard viewer.

6.6 Summary

Data transmission through digital store-and-forward communication transponders on board packet satellites, although generally offers a cheap and efficient means of communications, have some limitations. Thus, the present operations render the system far from ready for commercial scale services. However, it must be noted that these communication systems are experimental, having evolved through different experiments and experiences. Transmission of data at 9600 baud rate started officially in January 1992. The current problems and limitations will be addressed in future designs, with the help of advanced technology thus reducing the cost of many important components. The speed of downloading files is a function of the file size, number of users within the satellite footprints, the popularity of the file, and downlink signal strength. Very large files are not very much favoured by this system as they take more downloading time which may be unacceptable in practical applications. Similar arguments are applicable for data uploading speed. Although CCD images are of meteorological scale, they cannot be used in weather forecast for the two reasons, (1) the nature of their distribution is limited by the broadcast protocol system performance, and (2) Only the visible spectrum range is considered leaving other useful ranges of the spectrum. Therefore, for image data which are usually associated with large files to be transmitted through such a system, they must be compressed. Compression methods which can achieve as much compression ratio as possible without seriously affecting the quality of the image are most preferred. In doing so transmission time might be reduced to practically acceptable time and at the same time serving the memory space in the satellite server. The next chapter discusses various compression techniques which can be used to improve the system performance with respect to images transmission.

References

1. Sweeting M. N, et al, 1990. *University Of Surrey Launches Two New Satellites.* OSCAR NEWS, The Official J. Of AMSAT-UK For All Users Of Oscar Satellites, No. 81, pp 17-22, February.
2. Sweeting M. N, 1992. *UOSAT Microsatellite Missions.* Electronics and Communication Engineering Journal, pp 141-150, June.
3. Ward J and Harold Price, 1990. *Pacsat File Header Definition.* ARRL 9th Computer Networking Conference, pp 245-252, London, Ontario Canada, September, 22.
4. PB & PG User Documents.
5. Miller J, 1989. *9600 Baud Packet Radio Modem Design.* ARRL 7th Computer Networking Conference, pp 135-146, October.
6. Ward J, 1993. *UO-22 Sategates And Improved Performance.* OSCAR NEWS, The Official J. Of AMSAT-UK For All Users Of Oscar Satellites, No. 99 pp 18-20, February.

Chapter 7

IMAGE COMPRESSION

7.1 Introduction

Due to the nature of orbit mechanics in which LEO satellites such as OSCAR 22 operate, together with other factors as discussed previously, the time available to access the satellite server is limited. As a result, uploading and downloading big files such as image files is practically difficult and quite a frustrating business particularly in areas with many ground-stations. One of the best ways to reduce the uploading and downloading time is to compress image data files to a reasonable size prior to uploading. There are mainly two categories of image compression; the reversible and irreversible compression techniques with the later having advantage of achieving high compression ratio. This chapter describes two irreversible compression methods, the JPEG and FRACTAL compression techniques which are known to achieve higher compression ratio, as applied to still-continuous tone images.

7.2 Image Compression: An Overview

Digital representation of images usually requires a very large number of bits. A 640 x 400 pixel, 24 bits colour image offers a photographic quality display, but it will use up to 768 kilobytes of disc space. This may impose unacceptable costs for storage and transmission purposes. Besides, as discussed in chapter six, such large files will take a long time to upload or download. Digital image compression is a branch of image processing which addresses the problem of reducing the amount of data required to represent a digital image. Image compression requires that image data be encoded in such a way that the output of the encoding process produces a

coded form of data which occupies less bytes to represent the original data. These coded image data can then be decoded to allow a reconstruction of the original image [5].

Image compression techniques can be divided into two main categories on the basis of their ability to reconstruct from the coded image: (1) the irreversible or *'lossy'* compression and, (2) the reversible or *'lossless'* compression. Advantages of reversible compression techniques over irreversible ones are: (i) the original image can be perfectly reconstructed after coding (ii) the comparison of compression algorithms can be solely done on the basis of their efficiency, compared to irreversible techniques where subjective methods are used. The disadvantage of reversible techniques is that a smaller compression ratio is achieved than in irreversible techniques.

The use of either method depends on the application of the image under consideration. For instance, in storage applications, lossless coding techniques are preferred. In image transmission applications, irreversible compression techniques are preferred because of their higher compression ratio as long as the resulting images are acceptable for visual or machine analysis. In medical applications, lossless coding may be mandatory for two reasons; (i) *Legal regulations:* different countries have different regulations regarding original patient data, in some countries data are required to be kept for a number of years without tampering with any information contained in image data no matter how useless it is. (ii) *Post-processing:* the losses introduced during compression process may be enhanced by post processing operations applied to images [6].

Due to the poor performance of the packet satellite store-and-forward system with respect to image data, as discussed in chapter six, the objective is to minimise the transmission time required to transfer large files such as image files. In this case, attention is focused on *'lossy'* compression techniques as they achieve a higher compression ratio.

There are many *'lossy'* compression techniques in applications, most of which combine different classical coding techniques such Transform coding, Huffman coding, Vector quantization, Run-length coding etc. in order to obtain greater coding efficiency [13]. In this category, the internationally recognised open standard for still-continuous tone image compression is the JPEG, the work of ISO. A new lossy compression technique based on Fractal mathematics has been developed recently and is known as FRACTAL compression technique. This technique achieves image data redundancy by approximating an original image by a fractal code, which when iterated produces a sequence of images converging to a fractal approximation of the original image [3]. In this work, only two methods; the JPEG and FRACTAL compression methods, which have been found to achieve higher coding efficiency, are discussed and compared.

7.3 JPEG Compression Standards

The demand for more and better images, within the constraints of the existing storage and transmission technology, has prompted the development of a number of image compression standards. For still-continuous tone images, the internationally recognised compression standards are the JPEG standards [1]. Still-frame continuous tone images are digital images not in motion. Images referred to throughout this thesis, are real-world pictures, such as scanned photographs. Cartoons, line drawings, and other non-realistic images can be economically stored on computer disc and are not the subject of compression presented. These standards have been developed by the Joint Photographic Experts Group (JPEG) in collaboration with the CCITT and ISO. These standards combine different classical coding techniques; transform coding, quantization and variable length coding techniques, to achieve high compression ratios (Fig. 7.1).

In classical image compression, encoding processes involve three basic operations; mapping, quantization and entropy coding. The mapping operation de-correlates image data which are usually highly correlated

between adjacent pixels, to reduce interpixel redundancy so that the quantizer and coder in the next stage can be used more efficiently. De-correlating transforms such as DFT (Discrete Fourier Transform), DCT (Discrete Cosine Transform), DHT (Discrete Hartley Transform), and the WHT (Walsh-Hadamard Transform) are commonly applied in mapping operations [4]. JPEG standards rely on the use of DCT transform which has been found to have computational advantages over other transforms [2]. The quantizer rounds-off each mapped datum to one of a smaller number of possible values so that fewer code words with fewer bits are required. Quantization operation removes pyschovisual redundant data in an image. This process is irreversible and is the main source of error in the reconstruction process. However, errors can be made arbitrarily small by increasing the quantization accuracy. The entropy coding operation, achieve more compression *losslessly* by removing coding redundancy and assigning code words to each quantizer output. Coding techniques applied in image compression includes, Huffman coding, Arithmetic coding, Run-length coding and others.

The structure of the JPEG algorithms comprises four modes of operation [2]; **Sequential encoding** where each image component is encoded in a zig-zag scan, **Progressive encoding** where the image is encoded in multiple scans for real time transmission applications, **Lossless encoding** where image data encoded can be perfectly reconstructed and **Hierarchical encoding** where the image is encoded at multiple resolutions so that lower resolution versions may be accessed without first having to decompress the image at its full resolution. Sequential encoding is the most widely used operation among the four modes.

7.3.1 Sequential Encoding

The most popular and basic machinery mode for JPEG image compression standards is the Sequential encoding or sometimes referred to as Baseline system coding. Its algorithm structure is the most general form of compression for wide applications [1]. Data encoding and decoding

sequences applied by the JPEG compression algorithms are summarised by block diagrams in Fig. 7.1.

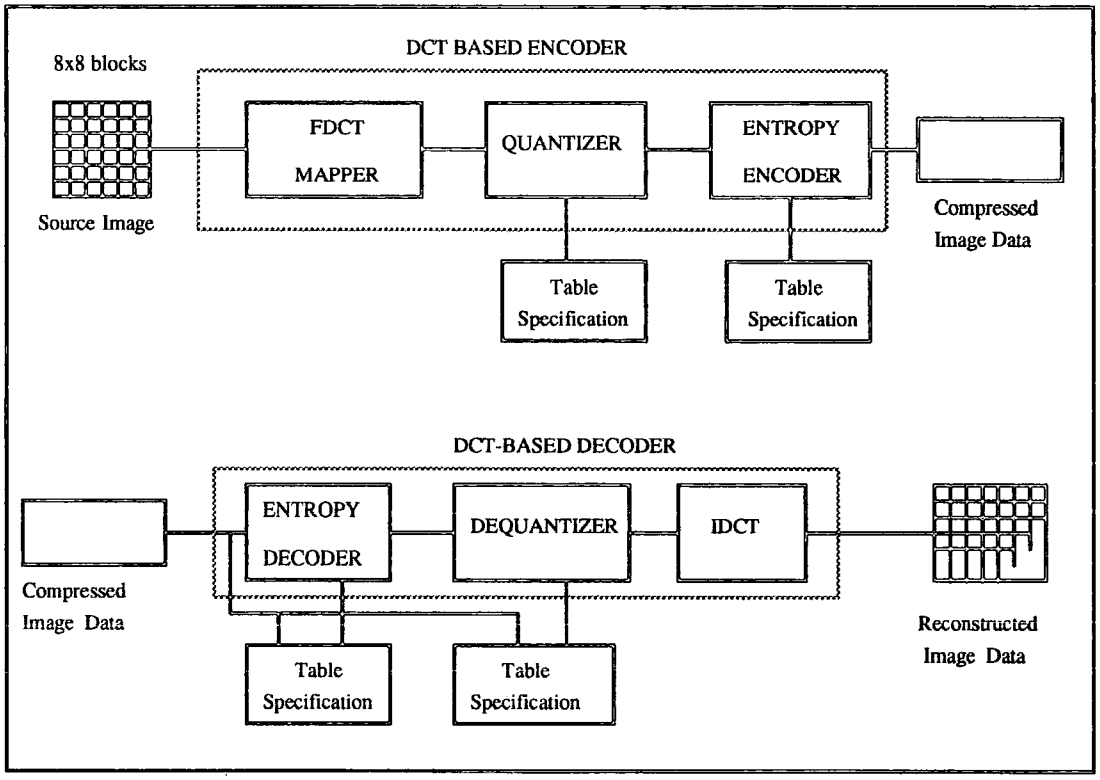

Fig. 7.1 DCT-Based Data Encoder - Decoder Processes In Block diagram [2].

7.3.1.1 DCT Mapper

The JPEG encoding process is done block wise, for a single component image the original image is sub-divided into pixel blocks of size 8 x 8 which are processed left to right, top to bottom. For every 8 x 8 block encountered, its 64 pixels are level shifted about zero by subtracting the quantity 2^{n-1}, where 2^n is the maximum number of grey levels. (for n = 8, the level shifting about zero will be shifting between -2^7 and 2^7-1). Each pixel is then transformed by a two dimension 8 x 8 FDCT (Forward Cosine Discrete Transform) to produce 64 coefficients. Coefficients with zero frequency in both dimensions are called the "DC coefficient" and the remaining 63 coefficients are called the "AC coefficients". In order to reduce the rather long duration for data processing by the DCT, a selected JPEG standard symmetrical definition of the Forward and Inverse DCT is

$$F(u,v) = \frac{1}{4}\left[C(u)C(v)\sum_{x}\sum_{y} f(x,y)\cos\left(\frac{(2x+1)u\pi}{16}\right)\cos\left(\frac{(2y+1)v\pi}{16}\right)\right] \quad \text{..... (7.1)}$$

$$f(x,y) = \frac{1}{4}\left[\sum_{u}\sum_{v} c(u)c(v)F(u,v)\cos\left(\frac{(2x+1)u\pi}{16}\right)\cos\left(\frac{(2y+1)v\pi}{16}\right)\right] \quad \text{.... (7.2)}$$

where C(u), C(v) = $\frac{1}{\sqrt{2}}$ for u, v = 0,

C(u), C(v) = 1 otherwise.

7.3.1.2 DCT Coefficients Quantizer

A quantizer is a device whose output can have only a limited number of possible values. Each input from the mapper's output is forced to one of the allowable output values. Suppose { t_j, j = 1, ... , L+1} defines a set of quantizer output coefficient levels. If u lies in the interval $(t_j, t_{j+1}]$, then u is mapped to finite set r_j, j∈[1, L], where L = 2^n, the number of grey levels. The quantity r_j, called the reconstruction level, is the quantized value of u and also lies in the interval $(t_j, t_{j+1}]$ [18]. The process reduces the accuracy of the mapper's output in accordance with some pre-defined fidelity criterion and is irreversible. Each of the 64 FDCT output coefficients is quantized independently by a uniform quantizer. This process is accomplished by use of user's selectable 64-elements quantization table where each quantization matrix is the quantizer step-size for its respective quantized coefficients (see Fig. 7.1). The output of the quantizer can be defined as

$$FQ(u,v) = \text{Integer Round}\ \frac{F(u,v)}{Q(u,v)} \quad \text{..... (7.3)}$$

where $FQ(u,v)$ = quantizer output,

$F(u,v)$ = FDCT mapper output,

$Q(u,v)$ = quantization table.

7.3.1.3 Entropy Encoding

The JPEG baseline entropy encoding is done in two steps. Prior to entropy encoding, the DC and AC coefficients from the quantizer output are treated separately. The DC term is encoded as a difference from the DC term of the previous block in the encoding order. The remaining 63 AC terms are not differentially encoded but are re-ordered into the zig-zag sequence placing low frequency coefficients before high frequency coefficients. The obtained sequence of symbols are then assigned variable-length codes. The recommended coding method for a baseline system is the standard Huffman coding. Arithmetic coding is not favoured by many implementors of the standards as it is a patented technique which cannot be used without paying a license fee and produces 5% more compression that Huffman coding.

7.4 Fractal Compression

The use of Fractals to simulate natural effects such as landscapes has become common. This has been achieved by using some deterministic algorithms, based on fractal theory of iterated transformations, that iteratively generate patterns resembling landscapes or any other natural effects [15]. Through experimentation, certain fractal codes can be developed, which will generate a pattern similar to landscape or any other natural effects. Fractal objects produced in this way, have the intrinsic property of having extremely high visual complexity while being very low in information content as they can be described and generated by these algorithms. Fractal compression is a new image compression technique which takes advantage of this property, where fractal transforms identifies the fractals that make up an image, hence finding the fractal formula that when iterated can recreate it to any desired degree of accuracy [7]. It is not the intention of this chapter to discuss in detail the mathematics of Fractal, rather relevant theories relating to image compression are presented.

7.4.1 Theory

The functions responsible to realise this goal are called affine transforms, which finally form a set of allowable contractive affine transforms forming Iteration Function System (IFS) codes. An IFS encodes an image by arranging it to be a unique fixed point of IFS, i.e. the image is mapped to itself. In simple English, an IFS encode an image by a process of shrinking, skewing, grey level scaling and rotation such that the image transformed visually matches the original image. To decode the image IFS are iterated from any starting image and the image will appear gradually as the iteration proceed [8].

An affine transformation ω: $\Re^2 \rightarrow \Re^2$ from two dimensional space $\Re^2$ into itself is defined by

$$\omega\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a_{11} a_{12} \\ a_{21} a_{22} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \quad \text{.....(7.4)}$$

where the a_{ij}'s and b_i's are real constants.

If we define A = matrix (a_{ij}), $\mathbb{b}$ = $(b_1, b_2)^T$, $\mathbb{a}$ = $(a_1, a_2)^T$, d distortion measure and I the original image to be encoded where T denotes matrix transpose, we can write

$$\omega(\mathbb{x}) = A\mathbb{x} + \mathbb{b} \quad \text{..... (7.5)}$$

The transformation ω is said to be contractive by scalar s if

$\exists$ s<1 such that $\forall$ $\mathbb{x}, \mathbb{y} \in \Re^2$, $d(\omega(\mathbb{x}), \omega(\mathbb{y})) \leq sd(x, y)$, and

d(I, ω(I)) is as small as possible, (app. fixed point) .....(7.6)

ω is said to be a fractal code if its information content is less than the original image. If ω is contractive, a sequence $\{I_n\}_{n\geq 0}$ can be constructed by

the iteration $I_{n+1} = \omega(I_n)$, which given I_o, converges to a unique fixed point I_ω, say, such that $I_\omega = \omega(I_\omega)$. Further, the sequence converges to the same number regardless of the starting number. Also, by Contraction mapping theorem [16] $\{I_n = \omega^n(I_o)\}_{n\geq 0}$ is also true. However, in reality it is not easy to find a fixed point of an image, rather an approximation to it can be found with the help of collage theorem [16]. Following the analysis of Jacquin [10], by the use of repeated application of the triangular inequality in $(\Re^2, d)$, and the use of contractivity of ω, it has been shown that for any image I_o and n positive integers;

$$d(I, \omega^n(I_o)) \leq \frac{1}{(1-s)}[d(I,\omega(I))] + s^n d(I, I_o) \quad \text{.....(7.7)}$$

From the last equation, it can be seen that the convergence of the sequence depends much the contractivity s being less than 1 and the distortion d being small. If ω satisfies the above conditions, it is said to be self-transformable. A collection of affine transforms with the above properties which can map a source image into a contracted image form IFS codes .

An IFS is a collection of contractive affine transformations. A two dimensional IFS consists of a set of N affine transforms, N being positive integers, denoted by $\{\omega_1, \omega_2, \omega_3, \ldots, \omega_N\}$ each taking $\Re^2$ into $\Re^2$, together with a set of probabilities $\{p_1, p_2, p_3, \ldots, p_N\}$, where each $p_i > 0$ and $\sum_{i=1}^{N} p_i = 1$, to determine its importance relative to the other transformations.
An example of IFS codes expressed in tabular form for a fern are shown in table 7.1. The IFS codes for a fern represented in this table, through deterministic or random iterated algorithms, will always produce a fern. The quality of the fern produced in this way will depend on the number of iterations specified.

TABLE 7.1

ω	a_{11}	a_{12}	a_{21}	a_{22}	b_1	b_2	p
ω_1	0.00	0.00	0.00	0.16	0.00	0.00	0.01
ω_2	0.20	-0.26	0.23	0.22	0.00	1.60	0.07
ω_3	-0.15	0.28	0.26	0.24	0.00	0.44	0.07
ω_4	0.85	0.04	-0.04	0.85	0.00	1.60	0.85

Fig. 7.1 A Table Showing IFS codes For a Fern [3].

7.4.2 Fractal Encoding Of Digital Images

In the above explanation, it has been shown how a set of contractive affine transforms, can be used to approximate an original image into IFS codes which when iterated visually matches the original image. As it turns out [11], this class of transformations is not sufficiently rich to describe an arbitrary image. Fractal coding on block basis is used so that the global transformation of the image into itself is then constructed from the combination of local block transformations.

Fractal block coding techniques for digital image proposed so far are; the ITT (Iterated Transform Technique), the patented system by Barnsley which is similar to ITT, and the BFT (Bath Fractal Transform) techniques [15]. In Jacquin [9] ITT approach, which has been implemented in fractal compression applications, fractal encoding process involves three basic steps; (1) the partitioning of the image, (2) the selection of distortion measure which will be used to measure the distortions in the cells and (3) the specification of a class of discrete contractive image transformations defined blockwise from which fractal codes are selected.

An original N x N digital image is partitioned into small non-overlapping square range of two different sizes; the larger cell (B x B) called the parent range cells and the smaller cell (B/2 x B/2) called child range cell. A parent range cell can be divided up to four child cells. Also, a set of overlapping

larger blocks which can occupy any position in the image known as domain blocks (D x D, D = 2B) are considered. To achieve fractal block encoding of an image, for every partition we must find a transformation from domain blocks to range blocks such that the transformed domain blocks are close approximations of the original range blocks. This involves two steps, the selection of a domain block (D x D) which will be contracted to a range block (B x B), then finding block transformations which will reduce distortions between the transformed domain and the original range.

A pool of domain blocks ($\wp_i$) from which different domain blocks will be selected, is created to cover all the defined range blocks in the image. For simplicity, these blocks are classified into four classes of blocks according to Ramamurthi and Gersho [17] classifications, shades, simple edge, mixed edge, and midrange blocks denoted by $\wp_s$, $\wp_{se}$, $\wp_{me}$, and $\wp_m$ respectively where $\wp_i = \wp_s \cup \wp_{se} \cup \wp_{me} \cup \wp_m$. Shades blocks are smooth blocks with no significant gradient, simple and mixed edges blocks are blocks with strong changes of intensity across the curve, and midrange blocks are blocks with moderate gradient but not definite edge.

Transformation of domain block to range block is a composition of two type of transformations, spatial contraction transforms (ς) which maps domain block into range block and block transforms (β) which process the contracted domain block to reduces distortion so that it is approximately the same as original range block. The later process affects the pixel values intensity by absorption at grey level, luminance shift, contrast scaling, and colour reversal transformations. A pool of all possible transformations of this nature is also created so that they can be easily selected whenever required. The domain and transformation pools together form what is termed as global pool ($\wp$ x ω).

It can be seen from the above process that encoding of a range block consists of finding the best pair of domain blocks in the pool of domains which, through spatial and block transformations, can visually match the range block. In mathematical terms the process can be summarised as follows. If I is restricted to the R_i range cell and D_i domain cell,

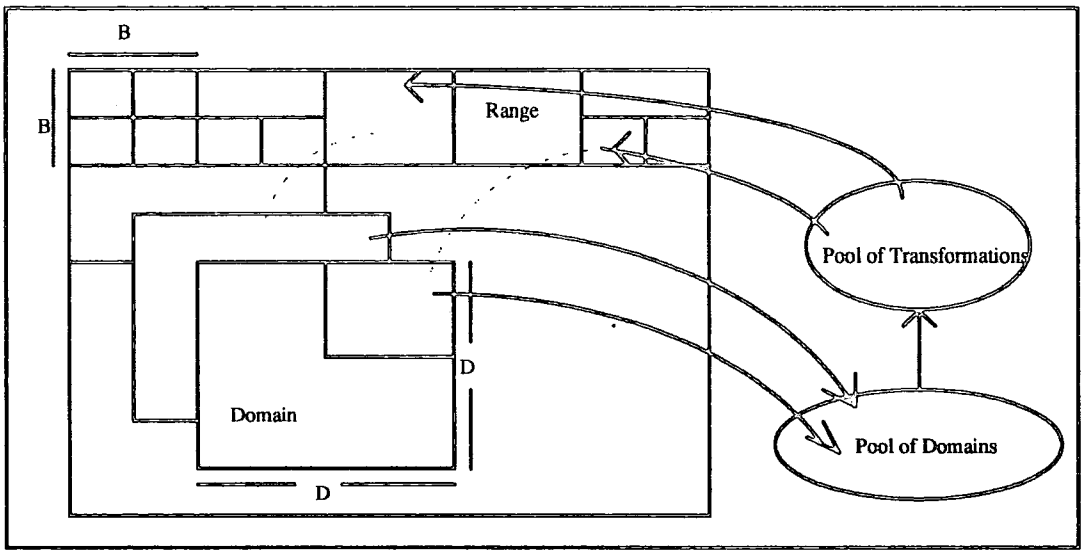

Fig. 7.3 Fractal encoding process of Image showing partitioning of image into range and domain blocks, and transformation of domain blocks to the range blocks through global pool.

$\forall (I) \in \Re^2$, $\omega(I) = \sum_{o}^{N-1} (\omega(I))_{R_i} = \sum_{o}^{N-1} \omega_i(I)_{D_i}$ where $\omega_i = \varsigma o \beta$ such that

the selection of best pair of domain block $(D_i, \varsigma_i) \in \wp \times \omega$ will result in distortion $d(I_{oRi}, \omega_i(I_{oDi}))$ to be minimum. This process has been summarised in fig. 7.4.

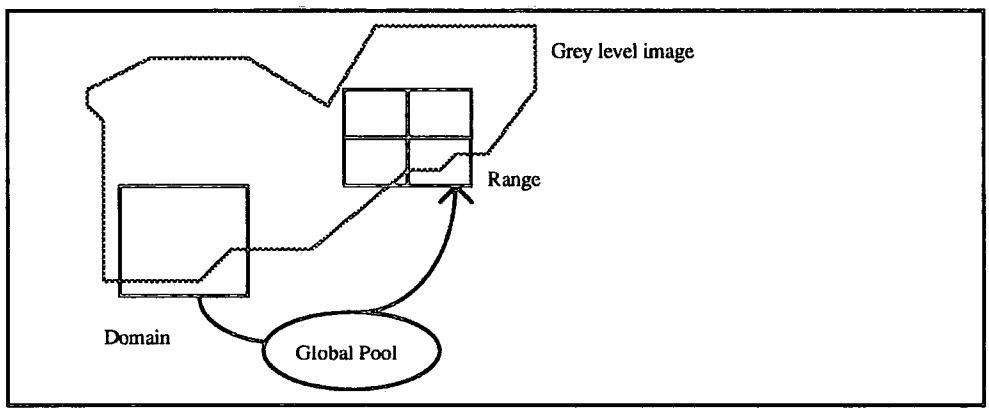

Fig. 7.4 Block matching for a 2-D grey scale image

The decoding process consist of iterating IFS codes in any initial image I_o, until convergence to a stable reconstructed image is obtained [8].

7.5 Summary

From the above discussion, it can be seen that both methods are computationally extensive. Fractal compression is more computationally intensive compared to JPEG. As a result, the speed of compression is affected. However, the speed of compression can be improved by using special compression hardware. The implementation of Fractal compression may be difficult particularly as it has been recently discovered and is proprietary compression technique where much information about how it works is hidden. The next chapter present results of compressing still-continuous tone images using both the JPEG and Fractal techniques followed by a discussion on the suitability of either method to image transmissions through computer networks such as Health-Net network.

Reference

1. Wallace G, 1991. *The JPEG Still Picture Compression Standards.* Communications of the ACM, Vol. 34 No. 4, pp 30-44, April.
2. Wallace G, 1990. *Overview Of The JPEG (ISO/CCITT) Still Image Compression Standards.* Proceedings Of Image Processing Algorithms And Techniques, SPIE, Vol. 1244, pp 220-232, February.
3. Barnsley F. M and Sloan D. A, 1988. *A Better Way To Compress Images.* BYTE, pp 215-223.
4. Rao K. R and Yip P, 1990. *Discrete Cosine Transform- Algorithms Advantages, Applications.* Academic Press, London.
5. Gonzalez R. C and Wood R. E, 1992. *Digital image Processing.* Academic Press.
6. Viergever A. M and Roos P, 1993. *Hierarchical Interpolation-An Effective Method For Reversible Compression Of Image.* IEEE J. Of Engineering In Medicine And Biology, pp 48-55, March.
7. Wright A, 1992. *FRACTALS Transform Image Compression.* Electronic World & Wireless World, pp 208-211, March.
8. Dettmer R, 1992. *Form And Function Fractal-Based Image Compression.* IEE Electronics Communications Review, pp 323-327, September.
9. Jacquin A. E, 1990. *A Novel Fractal Block-Coding Techniques For Digital Images.* IEEE International Conference on ASSP, pp 2225-2228.
10. Jacquin A. E, 1992. *Image Coding Based On A Fractal Theory Of Iterative Contractive Image Transformations.* IEEE Trans. On Image Processing, Vol. 1 No. 1, pp 18-30, January.
11. Waite J, 1990. *A Review Of Iterated Function Systems Theory For Image Compression.* IEE Colloquium On The Application Of Fractal Techniques In Image Processing, pp 1-8, 3rd, Dec.
12. Beaumont J. M, 1991. *Image Data Compression Using Fractal Techniques.* BT Tech. J Vol. 9 No. 4, pp 93-109, October.
13. Roos P, et al, 1988. *Reversible Intraframe Compression Of Medical Images.* IEEE Transactions On Medical Imaging, Vol. 7 No. 4, pp 328-336, December
14. Monro D. M, 1993. *Class Of Fractal Transforms.*

IEE Electronics Letters, Vol. 29 No. 4, pp 362-363, 18. February.
15. Peitgen H and Dietmer S (editors), 1988. The Science Of Fractal Images. Springer-Verlag, London.
16. Barnsley M,1988. *Fractals Everywhere.* Academic Press.
17. Ramamuthi B and Gersho A, 1986. *Classified Vector Quantization Of Images.* IEEE Trans. on Communications, Vol. COM-34 No. 11, November.
18. Jain A. K, 1981. *Image Data Compression: A Review* Proc. IEEE Vol. 69, March.

Chapter 8

IMAGE COMPRESSION COMPARATIVE RESULTS

8.1 Introduction

Irreversible compression methods achieve a higher compression ratio and are very useful in reducing the transmission time of image data through store-and-forward mail-box on-board packet satellites. There are different irreversible compression techniques with different coding efficiency as discussed in chapter seven. Since the objective is to find a compression technique which achieves higher compression ratio with minimum distortion to the reconstructed image to be used in the described network, a comparison of the Fractal and JPEG compression techniques as applied to still-continuous tone images suitable for digital store-and-forward mailbox system or Health-Net networks are discussed.

8.2 The JPEG Standards Implementation

The implementation of the JPEG standards is based on the sequential baseline system as described in the previous chapter. There are a lot of parameters to the JPEG compression-decompression process which results in different implementation of the standards by software vendors. For instance, as described before the entropy coding part demands Huffman or Arithmetic coding techniques to be used, and the quantizer demands that a quantization matrix table be supplied by user which can be defined by the implementor of the standards to suit their need. However, the JPEG standards have tried to specify quantization matrix tables for a wide range of applications and are commonly implemented, even though they are not standard by themselves. A public domain software from the Independent JPEG Groups (appendix 4) provides a good implementation of such standards. This software provides a wide

range of quantization matrix tables (about 100) which can be used to obtain different image compression ratios for a single image under consideration to suite many applications.

8.2.1 Image Quality Setting

JPEG compression involves a lot of parameters in the image encoding process. By adjusting these parameters, a trade off is made between compressed image size against reconstructed image quality. The quantizer should be supplied with quantization matrix table, whose nature determines the quality of the reconstructed images, the coarser the table the higher the compression ratio, but the poorer the quality of reconstructed image. Different implementors of the JPEG baseline standards are forced to leave the selection of the quantization table to users because the analysis of the reconstructed image quality is based on a subjective fidelity criterion evaluation. In this program, a hundred different tables have been specified, selected by adjusting the compressor's "quality" setting normally denoted by the letter Q ($0 \leq Q \leq 100$). This quality scale is purely arbitrary, and adjusting the Q values image quality can be traded off against file size. The lower the value of Q, the higher the compression ratio but the poorer the image quality. This setting will vary from one image to another and from one observer to another.

A test image, with original size $n_1 = 150843$ bytes has been used to determine the compression ratio and relative redundancy that can be achieved by different values of Q. Data redundancy, which is the measure of how successful the compression can be, is mathematically defined as follows [5]; If n_1 and n_2 denote the number of information carrying units in two data sets which represent the same information, the relative data redundancy R_D of the first data set can be defined as

$$R_D = \left(\frac{(C_R - 1)}{C_R} \right) \quad \text{..... (8.1)}$$

where C_R, commonly called the compression ratio is

$$C_R = \frac{n_1}{n_2} \quad \text{.....(8.2)}$$

These equations are fundamental in measuring the effect of any compression method. By recording compressed image size achieved for some selected quality setting Q, and using the above expressions, a table (TABLE 8.1) has been constructed which can be used to analyse the relationship between quality setting, compressed image size and the quality of the reconstructed image. Different values of compressed image size data (n_2) have been used to calculate the compression ratios and the relative redundancies for some selected Q values. From the table, graphs of compressed image size against Q values (Fig. 8.1), compression ratio against Q values (Fig 8.2), and relative redundancy in percentage against Q values (Fig. 8.3) have been plotted.

It can be seen from the graphs that lower values of Q, which are associated with coarse quantization tables, achieve higher compression ratios. From equation (8.1) the corresponding relative redundancies imply that a high percentage of original data is redundant. On the other hand, high values of Q achieve a lower compression ratio. For Q = 100, there is no effective compression rather there is expansion, for C_R values less than 1 which give a negative relative redundancy, which is not the desired result. However, the higher the Q value the better the image quality of the reconstructed image.

The quality of the decompressed image associated with these different values of Q can be seen in fig. 8.4. Using the subjective evaluation of image quality, which is simpler and more convenient mechanism for evaluating information loss for irreversible compression techniques, side by side comparisons of obtained images is done. It has been found that the general useful range of Q values is from 5 to 95. A simple absolute rating scale for image quality can be defined as follows; Excellent (Q = 95), highest useful quality setting producing image indistinguishable from the original image; Fine (Q = 75), usually set as a default value in many JPEG application programs, which produces high quality reconstructed image; Passable (Q = 50), produces slight defects but the image is of an acceptable quality; Marginal (Q = 30), produces images which is sometimes blocky, and is a typical characteristic of JPEG which

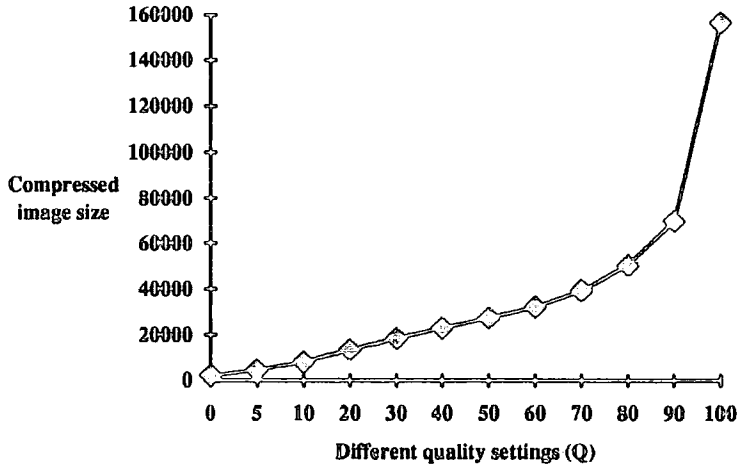

Fig. 8.1 Graph of Compressed image size against Quality setting.

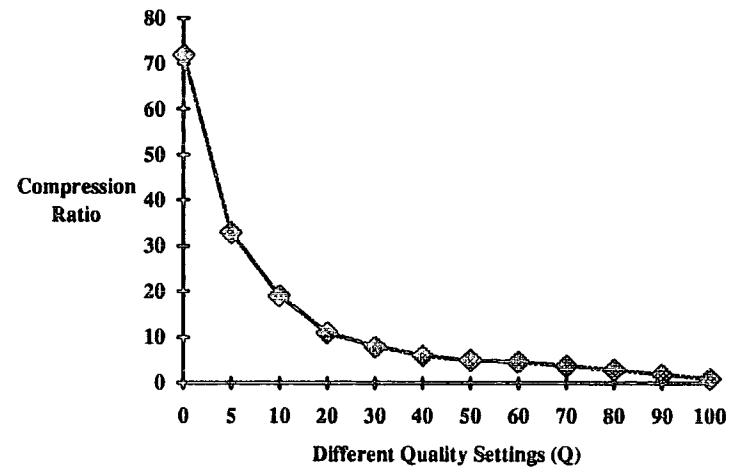

Fig. 8.2 Graph of Compression ratio against Quality setting.

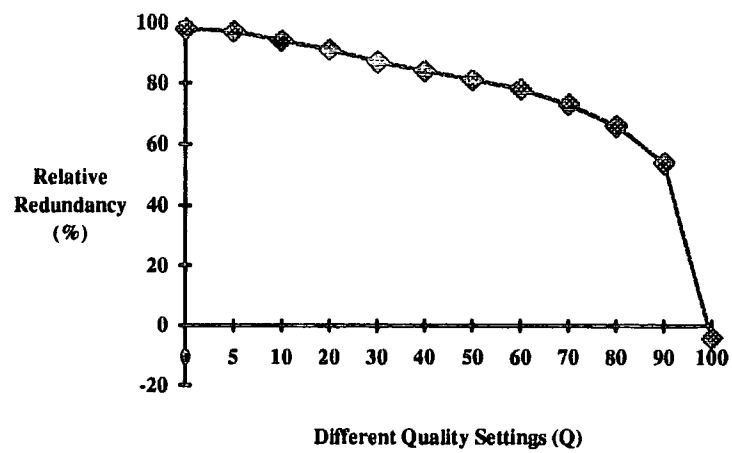

Fig. 8.3 Graph of relative redundancy against Q value settings.

will need improvement; Inferior (Q = 5), can be used only for previewing. However, best results, in terms of both compressed image size and image quality, are obtained for Q values between 50 - 95, in this range, depending on the image under consideration, the decompressed image will have minor visual differences from the original image. For instance in fig. 8.5, for Q = 95, the size of compressed image is approximately 162000 bytes less than the original size, the time required to transmit this image is reduced to approximately two thirds of the original time. Definitely further compression will reduce transmission time. The ability of retaining an acceptable quality of the reconstructed image while discarding a lot of visually insignificant data, will reduce the transmission time required for packet satellite operations, hence improving their performance.

TABLE 8.1 (n_1 = 150843)

Q VALUE	n_2	$C_R = \frac{n_2}{n_1}$	$R_D = 1 - \frac{1}{C_R}(\%)$
0	2100	72.0	98
5	4492	33.0	97
10	7898	19.0	94
20	13465	11.0	91
30	18628	8.0	87
40	23188	6.0	84
50	27560	5.0	81
60	32266	4.6	78
70	39454	3.8	73
80	50688	2.9	66
90	69973	2.0	54
100	156409	1.0	-4

8.3 Fractal Compression Implementation

The implementation of Fractal compression is computation intensive as explained previously. Further more, Fractal compression is a proprietary compression technique. As a result not much information is available as to how compression of still-continuous tone can be achieved using this technique. This disadvantage has made Fractal compression unpopular in many applications. However, a software 'POEM IMAGE INCORPORATED III -The Essential Desktop Publishing Tool' purchased from Iterated System, Inc. is used in this analysis.

The Image Incorporated-III software give a great range of compression, users are free to determine the size of the file to be compressed in contrast to JPEG where the file size is determined by the Q value setting. Therefore, the compression performance of this technique is analysed in a different way from the JPEG. However, for comparison purposes images are compressed to file sizes similar to those achieved by different JPEG quality 'Q' settings when used on the same image.

Image Incorporated-III, offer three methods of compressing image data, by using normal Fractal compression algorithms, Fractal Transform Templates (FFTs) to assist the Fractal algorithm in improving the quality of the decompressed image, and archiving format which can achieve 20% more compression or 80% more if used in conjunction with FFTs [9]. However, the most important method for comparison is by the Fractal compression algorithms. The other remaining two are merely assisting tools which contribute little to the efficiency of Fractal compression algorithms. Image compressed by Fractal compression algorithms can be seen in fig. 8.6.

8.4 JPEG And Fractal Comparisons

Comparisons between the JPEG and Fractal compression techniques, is discussed in this part based on their application suitability to still-continuous tone images and the store-and-forward e-mail network on

board pacsats, which is similar to the intended 'Health-Net' network application.

Both methods are lossy compression techniques, the reconstructed image is not the same as the original image due to information loss in the encoding-decoding processes as explained previously. One of the disadvantages of lossy compression techniques is that their performance depends on the nature of the original image, hence, assessing their efficiency is not solely based on their performance [7]. A convenient way of analysing their performance is by subjective fidelity criterion, where reconstructed image quality is measured by subjective evaluations of a human observer and may take the form of using an absolute rating scale or side-by-side comparisons of original and reconstructed images [2]. As a result, comparing the efficiency of the two techniques is difficulty unless confined to a particular application or original image nature. The convenient ways of comparing the two techniques is based on some fundamental comparisons factors in image data compression notably; implementation flexibility, coding efficiency, speed of compression-decompression processes, resolution dependency of the compressed file format, reconstructed image quality, and cost of obtaining the software.

Fractal compression implementation offers more flexibility than JPEG compression. As discussed before, implementation of JPEG differs depending on the chosen parameters such as quantization tables and entropy coding method at the coding stage. It is clear then that JPEG software can not be used for compression-decompression by either side unless there is an agreement to run the same software. Global usage of the JPEG compression may demand further agreement, hence hindering its flexibility. Fractal IFS codes can be used to reconstruct images without difficulty as they are just mathematical relations describing images, in other words any Fractal decoding software can decode IFS codes in its standard format.

The comparison based on compression power between the two techniques is the most important determining factor in deciding the suitability of the technique for Health-Net applications, as the objective is to use any method which can achieve a higher compression ratio in

order to reduce image transmission time. A 24-bit original image has been compressed using both techniques to observe the maximum compression ratio achieved by each technique with minimum distortion to the reconstructed image. It has been found that Fractal compression technique, using the POEM Image Incorporated-III, can achieve minimum compressed file size of 6,000 bytes with a poor reconstructed image. A similar compressed file size can be achieved by the JPEG compression technique using 20 as the quality setting. The reconstructed image quality is better than one produced by Fractal compression algorithms (Fig. 8.5).

Although the technique is claimed to achieve higher compression ratio [10], the compression ratio achieved by Fractal compression software in this observation casts doubt on the general efficiency of the Fractal compression algorithms, and lack of comparative evaluations has resulted in scepticism toward Fractal compression [8]. As a result, the compression achieved hardly justifies cost of buying the software. The use of Fractal technique in humanitarian services such as Health-Net is particularly not economically attractive to the developing countries. JPEG compression techniques are open standards, they can be used and implemented freely by any user, the JPEG compression software can be obtained free of charge from groups such as Independent JPEG group, or can be obtained at a relatively low cost. Owing to the JPEG cost-effectiveness, it is likely that many intended users of Health-Net will use the techniques, hence improving user interactions.

Fractal compression is claimed to give excellent reconstructed image quality [9]. However, the evaluation of the quality of the reconstructed image is subjective in lossy compression techniques. Different applications will require different image quality depending on the human observer as well, for instance in newspaper images, the quality of the image produced is not very critical, hence the original image can be compressed to as high compression ratio as possible. Lack of comparative standards for irreversible compression techniques and their dependency on the content of the original image leaves a lot of arguments floating [5]

Fractal compression offer greater decompression speed compared to JPEG standard [6]. But their compression speed is admitted by Barnsley [1] to be relatively slow due to computational intensive nature of finding image's fractal formulas, and hardware to speed up compression process has been made available. However, speed of decompression is not a serious factor in Health-Net network as it would have been in real-time communications.

The Fractal compression image format is resolution independent that is the compressed image are not limited by the resolution (size) of the original images from which they were compressed. A straight line in a Cartesian co-ordinate can be represented by the mathematical relation y = x, how long the line should be drawn will depend on how the values of x is specified. Similarly IFS codes are mathematical relations representing an image rather than bit-maps, decoding the image to the screen is resolution independent and will depend on preferred size of image without any file conversion. As a result the zooming power of Fractal compression algorithms is excellent. Fig 8.6 shows fractal decompressed images at two different resolutions, however, more resolution size can be obtained if higher graphic display adapter is used. Unfortunately JPEG does not offer this feature as their image formats are locked into one resolution (the number of pixels measured width x height).

8.5 Summary

The two compression techniques discussed in the previous chapter and in this chapter can both be used in image data compression. The efficiency of either technique is subjective as they are both lossy compression techniques which may also depend on the information content of the original image. However, the use of a cheap and open JPEG compression technique in humanitarian communication networks such as Health-Net is recommended. The compression ratio achieved by Fractal compression algorithms do not justify the cost of buying the software, and is economically unattractive, particularly to the developing countries. Further, JPEG standards being open standards offer more information on how its algorithms can achieve compression is free and

abundant, contrary to proprietary Fractal compression technique. However, Fractal compression being commissioned by Microsoft limited is likely to offer more user friendly implementation of the algorithms in conjunction with other advantages it has over JPEG standards, even though it will be at some cost. For commercial application, where the cost is secondary, Fractal compression can be used. The next chapter presents the conclusions of this research.

Fig. 8.4
COMPRESSED-DECOMPRESSED IMAGES FOR DIFFERENT QUALITY 'Q' SETTINGS

Original Image (24-bits True Colour 192054 bytes)

Q = 95 (30248 bytes Before Decompression)

Q = 20 (6835 bytes Before Decompression)

Q = 75 (17321 bytes Before Decompression)

Q = 50 (11753 bytes Before Decompression)

Q = 5 (3,158 bytes Before Decompression)

FIG. 8.5
COMPARISONS BETWEEN FRACTAL AND JPEG COMPRESSION TECHNIQUES

Original Image (192054 bytes).

Reconstructed JPEG Image (Q = 20, 6801 bytes compressed)

Reconstructed FRACTAL Image (6835 bytes compressed).

FIG. 8.6

FRACTAL DECOMPRESSED IMAGES

Fractal Decompressed Image At Half The Resolution 320 x 200

Fractal Decompressed Image At Full Resolution 640 x 400 (15133 bytes)

Reference

1. Barnsley F. M and Sloan D. A, 1988. *A Better Way To Compress Images.* BYTE, pp 215-223.
2. Gonzalez R. C and Wood R. E, 1992. *Digital image Processing.* Academic Press.
3. Viergever A. M and Roos P, 1993. *Hierarchical Interpolation-An Effective Method For Reversible Compression Of Image.* IEEE J. Of Engineering In Medicine And Biology, pp 48-55, March.
4. Wright A, 1992. *FRACTALS Transform Image Compression.* Electronic World & Wireless World, pp 208-211, March.
5. Dettmer R, 1992. *Form And Function Fractal-Based Image Compression.* IEE Electronics Communications Review, pp 323-327, September.
6. Waite J, 1990. *A Review Of Iterated Function Systems Theory For Image Compression.* IEE Colloquium On The Application Of Fractal Techniques In Image Processing, pp 1-8, 3rd, Dec.
7. Roos P, et al, 1988. *Reversible Intraframe Compression Of Medical Images.* IEEE Transactions On Medical Imaging, Vol. 7 No. 4, pp 328-336, December
8. Monro D. M, 1993. *Class Of Fractal Transforms.* IEE Electronics Letters, Vol. 29 No. 4, pp 362-363, 18. February.
9. POEM COLOBOX, 1993. User's Guide Version 2.0 Iterated Systems, Inc. Norcross Georgia.
10. POEM IMAGE INCORPORATED-III, 1993. User' Manual Iterated Systems, Inc. Norcross Georgia.

Chapter 9

CONCLUSION AND RECOMMENDATIONS

The use of amateur satellites for non real-time data communication networks promises a new generation of data communication networks as demonstrated in this research. A cost effective way of learning space science and related technologies which can easily involve schools, universities and independent organisations has emerged. The standardisation of communication protocols used in packet satellites will facilitate easy interactions among users of the services provided. "Health-Net" electronic mail network for humanitarian medical application is one example of services resulted from the amateur packet satellite digital store-and-forward communication technique. Achievements made in the Health-Net network have resulted in construction of a dedicated satellite for the service, known as HEALTHSAT2, to be launched soon. Since digital image technology has helped transforming microcomputers into visual communication tools, the use of this service for image data transmission became attractive. However, not much was known about their performances regarding digital image data transmission. This thesis has addressed the performance of PACSAT digital store-and-forward communication technique regarding still-continuous tone digital image data transmission.

Digital image files are associated with large files, the use of this kind of communication channel for digital image data transmission have some limitations. Due to the nature of LEO satellites orbit mechanics, and the speed of data transmission currently in use, time to access the satellite by the ground station is limited. Very large image files take a long time (several days for an image with file size of 300000 bytes) to upload or download, which may be unacceptable to users as discussed in chapter six. At the present situation, the only efficient way of transmitting image data through this channel is by compressing image files, the smaller the

image file the shorter the transmission time. *Lossy*, the JPEG and Fractal compression techniques, which are known to achieve higher compression ratios compared to *lossless* ones are the most favourable as they effectively reduce the transmission time.

This project compared the JPEG and Fractal compression techniques. Although Fractal compression technique offer a simple and fast decompression process, its compression process is computationally intensive, resulting in a slow compression process. The efficiency of compression algorithms developed so far do not produce convincing results for general applications as shown in chapter eight. Due to secrecy surrounding it and lack of comparative evaluations, many experts in image compression have some reservations. Further studies to improve the Fractal compression algorithms is required even though the technique is proprietary. The use of the JPEG open standards compression technique in still-continuous tone image data transmission for Health-Net applications is recommended, as the compression ratio achieved by Fractal compression technique do not justify the cost of buying the software, particularly when cost effective means of communication is the primary objective in many humanitarian organisations. However, Fractal compression technique can also be used as it has some advantages over the JPEG as explained in chapter eight, particularly to users who cost of obtaining the software is not a problem. Further, its copyright being licensed to Microsoft limited is likely to produce a more user friendly application environment at a reasonable price in future.

JPEG compression standards involves a lot of parameters resulting in different implementation of the standards as discussed in chapters seven and eight. While JPEG compression techniques can be easily used among few partners, global use of the JPEG compression technique will require further agreements, among users at international level, about the implementation of the JPEG standards. The use of this lossy compression technique in medical images will depend on the legal aspect governing application of such images in different countries as discussed in chapter seven and eight. Lossless compression techniques which are the alternative for medical images, are not helpful at the present situation as they do not reduce the transmission time effectively.

Although the efficiency of the channel in small scale applications is very good, their efficiency for large scale applications is limited as discussed in chapter six. At present, a large number of users within the satellite footprint tend to affect their performances as explained previously, hence such communication channel is still far away from commercial applications. However, for small scale applications such as Health-Net, the communication link will work efficiently as not many, probably one or two medical schools may be available within the satellite foot print. Further more, these satellites are experimental satellites whose present achievement have come out of evolutionary development, and with time, improvement in communication channel is expected. At present, in cases where there are many medical schools within the country, schools within a certain area will be served best if one of the school will act as a gateway, commonly referred to as satgates, where only one station within the defined area is linked to the serve, hence reducing users congestion during the satellite pass. Other schools can be connected to the gateway station through different ways such as terrestrial PBBS and telephone BBS as discussed in chapter one and two.

Therefore, digital image transmission through the present packet satellite digital store-and-forward communication system, which is similar to the intended 'Health-Net' network, is useful if the images are compressed using lossy techniques. For Health-Net applications, JPEG compression techniques are recommended for economical reasons, as compression ratio achieved by Fractal compression technique can be achieved by JPEG as well at no cost. However, although expensive the Fractal compression technique has more advantages, apart from its compression power, and is promising a more user friendly implementation in future.

APPENDIX 1A

An Example Of Kepler Elements in NASA Format Downloaded From OSCAR 23

SB KEPS @ AMSAT $ORBS-023.N
2Line Orbital Elements 023.AMSAT

HR AMSAT ORBITAL ELEMENTS FOR AMATEUR SATELLITES IN NASA FORMAT
FROM N3FKV HEWITT, TX January 23, 1993 BID:$ORBS-023.N

DECODE 2-LINE ELSETS WITH THE FOLLOWING KEY:

1 AAAAAU 00 0 0 BBBBB.BBBBBBBB .CCCCCCCC 00000-0 00000-0 0 DDDZ
2 AAAAA EEE.EEEE FFF.FFFF GGGGGGG HHH.HHHH III.IIII JJ.JJJJJJJKKKKKZ
KEY: A-CATALOGNUM B-EPOCHTIME C-DECAY D-ELSETNUM E-INCLINATION F-RAAN
G-ECCENTRICITY H-ARGPERIGEE I-MNANOM J-MNMOTION K-ORBITNUM Z-CHECKSUM

TO ALL RADIO AMATEURS BT

UO-11
1 14781U 84 21 B 93018.09801447 .00000562 00000-0 10396-3 0 3955
2 14781 97.8270 51.0366 0012880 112.8535 247.4027 14.68829707474688

UO-14
1 20437U 90 5 B 93020.20159941 .00000166 00000-0 72468-4 0 7122
2 20437 98.6276 106.0754 0010598 258.9011 101.0980 14.29727708156286

AO-16
1 20439U 90 5 D 93018.24844793 .00000158 00000-0 69216-4 0 5419
2 20439 98.6325 104.8939 0010961 268.7027 91.2899 14.29787375156010

DO-17
1 20440U 90 5 E 93016.22272524 .00000178 00000-0 76914-4 0 5431
2 20440 98.6315 103.0533 0010830 273.4883 86.5060 14.29918799155731

WO-18
1 20441U 90 5 F 93016.75008970 .00000158 00000-0 69080-4 0 5458
2 20441 98.6315 103.6118 0011400 272.0913 87.8964 14.29903662155819

LO-19
1 20442U 90 5 G 93018.72050830 .00000170 00000-0 73697-4 0 5420
2 20442 98.6332 105.7156 0011996 266.5060 93.4752 14.29991574156105

UO-22
1 21575U 91 50 B 93021.72455528 .00000204 00000-0 76046-4 0 2411
2 21575 98.4886 100.1074 0008391 19.7776 340.3734 14.36767856 79638

KO-23
1 22077U 92 52 B 93006.08586143 -.00000000 00000-0 99999-4 0 866
2 22077 66.0809 303.5860 0013347 229.3565 130.6278 12.86275910 18999

APPENDIX 1B

An Example Of Kepler Elements in AMSAT Foramt Downloaded From OSCAR 23

SB KEPS @ AMSAT $ORBS-046.O
Orbital Elements 046.OSCAR

HR AMSAT ORBITAL ELEMENTS FOR THE MICROSATS
FROM N3FKV HEWITT, TX February 15, 1993 BID:$ORBS-046.O
TO ALL RADIO AMATEURS BT

Satellite: UO-22
Catalog number: 21575
Epoch time: 93040.24894820
Element set: 243
Inclination: 98.4864 deg
RA of node: 118.3602 deg
Eccentricity: 0.0007834
Arg of perigee: 324.9777 deg
Mean anomaly: 35.0893 deg
Mean motion: 14.36774641 rev/day
Decay rate: 2.25e-06 rev/day^2
Epoch rev: 8229
Checksum: 318

Satellite: KO-23
Catalog number: 22077
Epoch time: 93006.08586143
Element set: 86
Inclination: 66.0809 deg
RA of node: 303.5860 deg
Eccentricity: 0.0013347
Arg of perigee: 229.3565 deg
Mean anomaly: 130.6278 deg
Mean motion: 12.86275910 rev/day
Decay rate: -.00000000 rev/day^2
Epoch rev: 1899
Checksum: 293

APPENDIX 2

An example of Downloaded Bulletin News File From OSCAR 23 All Headers Have Been Removed

SB SAT @ AMSAT $ANS-044.01
NTBP MISSION 2 SUCCESS

HR AMSAT NEWS SERVICE BULLETIN 044.01 FROM AMSAT HQ
SILVER SPRING, MD FEBRUARY 13, 1993 BID: $ANS-044.01
TO ALL RADIO AMATEURS BT

NORTH TEXAS BALLOON PROJECT SECOND LAUNCH A SUCCESS (...FINALLY!)

The launch of the second mission of the North Texas Balloon Project was a success on 6-Feb-93 (after December and January launch scrubs due to bad weather). Lift off was at 15:09 UTC and landing occurred at 16:55 UTC. The payload was recovered less than an hour later just a few hundred yards from the shores of Lake Whitney, only 32.5 miles south of the launch site. Recovery time was excellent considering the payload actually landed 30 miles away from the predicted landing point!

We would like to thank all of you who participated in the launch by providing reception reports via the 40M launch net to Keith Pugh (W5IU).

If you would like a QSL card for Mission #2, please send your decoded telemetry and an SASE to Doug Howard (KG5OA), 2517 Coldstream Drive, Fort Worth, TX 76123.

Our next launch will require only slight modification to the payload and should be ready for launch in the next few months. So stay tuned to this bulletin for more information on Mission #3!

[The AMSAT News Service (ANS) would like to thank Doug Howard (KG5OA) for this bulletin item.]

APPENDIX 3

To: G6ZRU

Thanks for your message. I am helping a student working on pacsat at Durham University. I am aware that the University is user member of AMSAT-UK with the number 3513.92, ext. 93 soon.

Thanks, G1YNA, Matts.

Uploader G6ZRU
Source G6ZRU
Destination G1YNA
Filename HELLO AGAIN
Tittle USEFUL?

To: G1YNA
From: G6ZRU
Time: 123639 UTC
Date: 24MAR93

Hello,

I hope the programs that you downloaded are useful. You may not be aware that all amateur satellites are provided and maintained by contributions from individual amateurs. They are NOT funded by national societies. Further details about all amateur satellites and information about how to contribute may be obtained by sending an SASE to: AMSAT-UK,LONDON E12 5EQ.

73's from Fred G6ZRU, AUK #2632
@ UO22,KO23,GB7VRB or Compuserve 100024,2534.

APPENDIX 4A

Source Codes For JPEG Compression Program

```
/*
 * jcmain.c
 *
 * Copyright (C) 1991, 1992, Thomas G. Lane.
 * This file is part of the Independent JPEG Group's software.
 * For conditions of distribution and use, see the accompanying README file.
 *
 * This file contains an MS-DOS user interface for the JPEG compressor.
 * It follows DOS practice more closely than does the standard jcmain.c.
 * This compiles under Borland C or Microsoft C; don't know about others.
 * If you are using jmemnobs.c, #define FREE_MEM_ESTIMATE=0.
 */

#include "jinclude.h"
#include <stdlib.h>             /* to declare exit() */
#include <string.h>             /* string functions */
#include <signal.h>             /* to declare signal() */

#ifdef __TURBOC__
                                /* Borland C declarations */
#include <dir.h>                /* for findfirst/findnext */
typedef struct ffblk FF_DATA;
#define FF_NAME  ff_name
#define FINDFIRST(path,blk)         findfirst(path, blk, 0)
#define FINDNEXT(blk)               findnext(blk)

#include <alloc.h>              /* for farcoreleft() */
/* In Borland C, force -m equal to 95% of farcoreleft figure */
#define FREE_MEM_ESTIMATE  ((farcoreleft() * 95L) / 100L)

#else /* !__TURBOC__ */
                                /* Microsoft C declarations */
#include <dos.h>                /* for findfirst/findnext */
typedef struct find_t FF_DATA;
```

```
#define FF_NAME  name
#define FINDFIRST(path,blk)         _dos_findfirst(path, _A_NORMAL, blk)
#define FINDNEXT(blk)               _dos_findnext(blk)

#endif /* __TURBOC__ */

#define READ_BINARY     "rb"
#define WRITE_BINARY    "wb"

#ifndef EXIT_FAILURE                /* define exit() codes if not provided */
#define EXIT_FAILURE  1
#endif
#ifndef EXIT_SUCCESS
#define EXIT_SUCCESS  0
#endif


#include "jversion.h"           /* for version message */


/*
 * PD version of getopt(3).
 */

#include "egetopt.c"


/*
 * This routine determines what format the input file is,
 * and selects the appropriate input-reading module.
 *
 * To determine which family of input formats the file belongs to,
 * we may look only at the first byte of the file, since C does not
 * guarantee that more than one character can be pushed back with ungetc.
 * Looking at additional bytes would require one of these approaches:
 *     1) assume we can fseek() the input file (fails for piped input);
 *     2) assume we can push back more than one character (works in
 *        some C implementations, but unportable);
```

```
 *    3) provide our own buffering as is done in djpeg (breaks input readers
 *       that want to use stdio directly, such as the RLE library);
 * or  4) don't put back the data, and modify the input_init methods to assume
 *       they start reading after the start of file (also breaks RLE library).
 * #1 is attractive for MS-DOS but is untenable on Unix.
 *
 * The most portable solution for file types that can't be identified by their
 * first byte is to make the user tell us what they are.  This is also the
 * only approach for "raw" file types that contain only arbitrary values.
 * We presently apply this method for Targa files.  Most of the time Targa
 * files start with 0x00, so we recognize that case.  Potentially, however,
 * a Targa file could start with any byte value (byte 0 is the length of the
 * seldom-used ID field), so we accept a -T switch to force Targa input mode.
 */

static boolean is_targa;        /* records user -T switch */


LOCAL void
select_file_type (compress_info_ptr cinfo)
{
  int c;

  if (is_targa) {
#ifdef TARGA_SUPPORTED
    jselrtarga(cinfo);
#else
    ERREXIT(cinfo->emethods, "Targa support was not compiled");
#endif
    return;
  }

  if ((c = getc(cinfo->input_file)) == EOF)
    ERREXIT(cinfo->emethods, "Empty input file");

  switch (c) {
#ifdef GIF_SUPPORTED
  case 'G':
```

```
    jselrgif(cinfo);
    break;
#endif
#ifdef PPM_SUPPORTED
  case 'P':
    jselrppm(cinfo);
    break;
#endif
#ifdef RLE_SUPPORTED
  case 'R':
    jselrrle(cinfo);
    break;
#endif
#ifdef TARGA_SUPPORTED
  case 0x00:
    jselrtarga(cinfo);
    break;
#endif
  default:
#ifdef TARGA_SUPPORTED
    ERREXIT(cinfo->emethods, "Unrecognized input file format --- if it's Targa, use -
T");
#else
    ERREXIT(cinfo->emethods, "Unrecognized input file format");
#endif
    break;
  }

  if (ungetc(c, cinfo->input_file) == EOF)
    ERREXIT(cinfo->emethods, "ungetc failed");
}


/*
 * This routine gets control after the input file header has been read.
 * It must determine what output JPEG file format is to be written,
 * and make any other compression parameter changes that are desirable.
 */
```

```
METHODDEF void
c_ui_method_selection (compress_info_ptr cinfo)
{
 /* If the input is gray scale, generate a monochrome JPEG file. */
 if (cinfo->in_color_space == CS_GRAYSCALE)
  j_monochrome_default(cinfo);
 /* For now, always select JFIF output format. */
#ifdef JFIF_SUPPORTED
 jselwjfif(cinfo);
#else
 You shoulda defined JFIF_SUPPORTED.  /* deliberate syntax error */
#endif
}


/*
 * Signal catcher to ensure that temporary files are removed before aborting.
 * Doesn't do anything during intervals where emethods is NULL.
 */

static external_methods_ptr emethods; /* for access to free_all */

LOCAL void
signal_catcher (int signum)
{
 if (emethods != NULL) {
  emethods->trace_level = 0; /* turn off trace output */
  (*emethods->free_all) ();   /* clean up memory allocation & temp files */
 }
 exit(EXIT_FAILURE);
}


LOCAL void
usage (void)
/* complain about bad command line */
{
```

```
  fprintf(stderr, "Usage: cjpeg [switches] inputfile(s)\n");
  fprintf(stderr, "List of input files may use wildcards (* and ?)\n");
  fprintf(stderr, "If input file name has no extension, .GIF is assumed\n");
  fprintf(stderr, "Output filename is same as input filename, but extension .JPG\n");
  fprintf(stderr, "Optional switches:\n");
  fprintf(stderr, "  -Q quality  Image quality (0..100; 5-95 is useful range)\n");
  fprintf(stderr, "  -o          Optimize Huffman table (smaller file, but slow)\n");
#ifdef TARGA_SUPPORTED
  fprintf(stderr, "  -T          Input file is Targa format (usually not needed)\n");
#endif
#ifndef FREE_MEM_ESTIMATE
  fprintf(stderr, "  -m memory   Maximum memory to use (default 300K); see
USAGE\n");
#endif
  fprintf(stderr, "  -d          Generate debug output\n");
  exit(EXIT_FAILURE);
}


/* Display progress report */

METHODDEF void
progress_monitor (compress_info_ptr cinfo, long loopcounter, long looplimit)
{
  if (cinfo->total_passes > 1) {
    fprintf(stderr, "\rPass %d/%d: %3d%% ",
            cinfo->completed_passes+1, cinfo->total_passes,
            (int) (loopcounter*100L/looplimit));
  } else {
    fprintf(stderr, "\r %3d%% ",
            (int) (loopcounter*100L/looplimit));
  }
  fflush(stderr);
}


/*
 * Check for overwrite of an existing file; clear it with user
```

```
 */

LOCAL boolean
is_write_ok (char * outfilename)
{
  FILE * ofile;
  int ch;

  ofile = fopen(outfilename, READ_BINARY);
  if (ofile == NULL)
    return TRUE;                    /* not present */
  fclose(ofile);            /* oops, it is present */

  for (;;) {
    fprintf(stderr, "%s already exists, overwrite it? [y/n] ",
            outfilename);
    fflush(stderr);
    ch = getc(stdin);
    fflush(stdin);                  /* flush rest of line */
    switch (ch) {
    case 'Y':
    case 'y':
      return TRUE;
    case 'N':
    case 'n':
      return FALSE;
    /* otherwise, ask again */
    }
  }
}


/*
 * Save current switch values here --- necessary for multiple input files!
 */

static int cur_quality;           /* -Q  */
static boolean cur_opt;                   /* -o */
```

```
static long cur_mem;		/* -m, or value of FREE_MEM_ESTIMATE */
static int cur_trace;		/* -d */

#define MAX_FNAME_LEN 150


/*
 * Process a single input file
 */

LOCAL void
process_one_file (char * infilename)
{
  struct compress_info_struct cinfo;
  struct compress_methods_struct c_methods;
  struct external_methods_struct e_methods;
  char outfilename[MAX_FNAME_LEN];
  int i;

  /* Select the input and output files */

  if ((cinfo.input_file = fopen(infilename, READ_BINARY)) == NULL) {
    fprintf(stderr, "cjpeg: can't open %s\n", infilename);
    return;
  }
  /* Make outfilename be infilename with .JPG substituted for extension */
  strcpy(outfilename, infilename);
  for (i = strlen(outfilename)-1; i >= 0; i--) {
    switch (outfilename[i]) {
    case ':':
    case '/':
    case '\\':
      i = 0;			/* stop scanning */
      break;
    case '.':
      outfilename[i] = '\0';	/* lop off existing extension */
      i = 0;			/* stop scanning */
      break;
```

```
  default:
    break;                      /* keep scanning */
  }
}
strcat(outfilename, ".JPG");

if (! is_write_ok(outfilename)) {
  fclose(cinfo.input_file);
  return;
}
if ((cinfo.output_file = fopen(outfilename, WRITE_BINARY)) == NULL) {
  fprintf(stderr, "cjpeg: can't create %s\n", outfilename);
  fclose(cinfo.input_file);
  return;
}

/* Initialize the system-dependent method pointers. */
cinfo.methods = &c_methods;
cinfo.emethods = &e_methods;
jselerror(&e_methods);        /* error/trace message routines */
jselmemmgr(&e_methods);  /* memory allocation routines */
c_methods.c_ui_method_selection = c_ui_method_selection;

/* Now OK to enable signal catcher. */
emethods = &e_methods;

/* Set up JPEG parameters. */
j_c_defaults(&cinfo, cur_quality, FALSE);
cinfo.optimize_coding = cur_opt;
e_methods.max_memory_to_use = cur_mem;
e_methods.trace_level = cur_trace;

/* Start up progress display, unless trace output is on */
fprintf(stderr, "Compressing %s => %s\n", infilename, outfilename);
if (cur_trace == 0)
  c_methods.progress_monitor = progress_monitor;

/* Figure out the input file format, and set up to read it. */
```

```
  select_file_type(&cinfo);

  /* Do it to it! */
  jpeg_compress(&cinfo);

  /* Clear away progress display */
  if (cur_trace == 0)
    fprintf(stderr, "\r                \r");
  fflush(stderr);

  /* All done. Close files, disable signal catcher */
  fclose(cinfo.input_file);
  fclose(cinfo.output_file);
  emethods = NULL;
}


/*
 * Process one filespec; expand wildcards
 */

LOCAL void
process_filespec (char * filespec)
{
  FF_DATA ffblock;
  char infilename[MAX_FNAME_LEN];
  int pathlen, i;
  boolean hasext;

  /* Determine whether there is a path and an extension */
  pathlen = 0;
  hasext = FALSE;
  for (i = strlen(filespec)-1; i >= 0; i--) {
    switch (filespec[i]) {
    case ':':
    case '/':
    case \\':
      pathlen = i+1;                /* path part ends here */
```

```
    i = 0;                      /* stop scanning */
    break;
   case '.':
    hasext = TRUE;              /* there is an extension */
    break;
   default:
    break;                      /* keep scanning */
   }
  }

  /* Make infilename be filespec; if no explicit extension, attach .GIF */
  strcpy(infilename, filespec);
  if (! hasext)
   strcat(infilename, ".GIF");

  /* Scan over matching files */
  if (FINDFIRST(infilename, &ffblock) != 0) {
   fprintf(stderr, "No files matching %s\n", infilename);
  } else {
   do {
    strcpy(infilename, filespec);
    strcpy(infilename + pathlen, ffblock.FF_NAME);
    process_one_file(infilename);
   } while (FINDNEXT(&ffblock) == 0);
  }
}


/*
 * The main program.
 */

GLOBAL int
main (int argc, char **argv)
{
  int c;

  /* Set up signal catcher. */
```

```
  emethods = NULL;
  signal(SIGINT, signal_catcher);
#ifdef SIGTERM                  /* not all systems have SIGTERM */
  signal(SIGTERM, signal_catcher);
#endif

  /* Scan command line, process switches */

  is_targa = FALSE;             /* initialize default switch values */
  cur_quality = 75;
  cur_opt = FALSE;
#ifdef FREE_MEM_ESTIMATE
  cur_mem = FREE_MEM_ESTIMATE;
#else
  cur_mem = 300000L;            /* reasonable default for DOS */
#endif
  cur_trace = 0;

  while ((c = egetopt(argc, argv, "Q:Tom:d")) != EOF)
    switch (c) {
    case 'Q':                   /* Quality factor. */
      if (optarg == NULL)
          usage();
      if (sscanf(optarg, "%d", &cur_quality) != 1)
          usage();
      break;
    case 'T':                   /* Input file is Targa format. */
      is_targa = TRUE;
      break;
    case 'o':                   /* Enable entropy parm optimization. */
#ifdef ENTROPY_OPT_SUPPORTED
      cur_opt = TRUE;
#else
      fprintf(stderr, "cjpeg: sorry, entropy optimization was not compiled\n");
      exit(EXIT_FAILURE);
#endif
      break;
    case 'm':                   /* Maximum memory in Kb (or Mb with 'm'). */
```

```
    { long lval;
        char ch = 'x';

        if (optarg == NULL)
          usage();
        if (sscanf(optarg, "%ld%c", &lval, &ch) < 1)
          usage();
        if (ch == 'm' || ch == 'M')
          lval *= 1000L;
        cur_mem = lval * 1000L;
    }
    break;
  case 'd':                     /* Debugging. */
    /* On first -d, print version identification */
    if (cur_trace == 0)
        fprintf(stderr, "Independent JPEG Group's CJPEG, version %s\n%s\n",
                JVERSION, JCOPYRIGHT);
    cur_trace++;
    break;
  case '?':
  default:
    usage();
    break;
  }

 /* Process file specifications */
 if (optind >= argc)
  usage();                      /* no filespecs?? */

 while (optind < argc)
  process_filespec(argv[optind++]);

 /* All done. */
 exit(EXIT_SUCCESS);
 return 0;                      /* suppress no-return-value warnings */
}
```

APPENDIX 4B

Source Codes For JPEG Decompression Program

```
/*
 * jdmain.c
 *
 * Copyright (C) 1991, 1992, Thomas G. Lane.
 * This file is part of the Independent JPEG Group's software.
 * For conditions of distribution and use, see the accompanying README file.
 *
 * This file contains an MS-DOS user interface for the JPEG decompressor.
 * It follows DOS practice more closely than does the standard jdmain.c.
 * This compiles under Borland C or Microsoft C; don't know about others.
 * If you are using jmemnobs.c, #define FREE_MEM_ESTIMATE=0.
 */

#include "jinclude.h"
#include <stdlib.h>             /* to declare exit() */
#include <string.h>             /* string functions */
#include <signal.h>             /* to declare signal() */

#ifdef __TURBOC__
                                /* Borland C declarations */
#include <dir.h>                /* for findfirst/findnext */
typedef struct ffblk FF_DATA;
#define FF_NAME  ff_name
#define FINDFIRST(path,blk)         findfirst(path, blk, 0)
#define FINDNEXT(blk)               findnext(blk)

#include <alloc.h>              /* for farcoreleft() */
/* In Borland C, force -m equal to 95% of farcoreleft figure */
#define FREE_MEM_ESTIMATE  ((farcoreleft() * 95L) / 100L)

#else /* !__TURBOC__ */
                                /* Microsoft C declarations */
#include <dos.h>                /* for findfirst/findnext */
typedef struct find_t FF_DATA;
```

```
#define FF_NAME  name
#define FINDFIRST(path,blk)         _dos_findfirst(path, _A_NORMAL, blk)
#define FINDNEXT(blk)               _dos_findnext(blk)

#endif /* __TURBOC__ */

#define READ_BINARY     "rb"
#define WRITE_BINARY    "wb"

#ifndef EXIT_FAILURE                /* define exit() codes if not provided */
#define EXIT_FAILURE  1
#endif
#ifndef EXIT_SUCCESS
#define EXIT_SUCCESS  0
#endif


#include "jversion.h"           /* for version message */


/*
 * PD version of getopt(3).
 */

#include "egetopt.c"


/*
 * This list defines the known output image formats
 * (not all of which need be supported by a given version).
 * You can change the default output format by defining DEFAULT_FMT;
 * indeed, you had better do so if you undefine GIF_SUPPORTED.
 */

typedef enum {
        FMT_GIF,            /* GIF format */
        FMT_PPM,            /* PPM/PGM (PBMPLUS formats) */
        FMT_RLE,            /* RLE format */
```

```
        FMT_TARGA,              /* Targa format */
        FMT_TIFF            /* TIFF format */
} IMAGE_FORMATS;

#ifndef DEFAULT_FMT             /* so can override from CFLAGS in Makefile */
#define DEFAULT_FMT     FMT_GIF
#endif

static IMAGE_FORMATS requested_fmt;


/*
 * This routine gets control after the input file header has been read.
 * It must determine what output file format is to be written,
 * and make any other decompression parameter changes that are desirable.
 */

METHODDEF void
d_ui_method_selection (decompress_info_ptr cinfo)
{
  /* if grayscale or CMYK input, force similar output; */
  /* else leave the output colorspace as set by options. */
  if (cinfo->jpeg_color_space == CS_GRAYSCALE)
    cinfo->out_color_space = CS_GRAYSCALE;
  else if (cinfo->jpeg_color_space == CS_CMYK)
    cinfo->out_color_space = CS_CMYK;

  /* select output file format */
  /* Note: jselwxxx routine may make additional parameter changes,
   * such as forcing color quantization if it's a colormapped format.
   */
  switch (requested_fmt) {
#ifdef GIF_SUPPORTED
  case FMT_GIF:
    jselwgif(cinfo);
    break;
#endif
#ifdef PPM_SUPPORTED
```

```
  case FMT_PPM:
    jselwppm(cinfo);
    break;
#endif
#ifdef RLE_SUPPORTED
  case FMT_RLE:
    jselwrle(cinfo);
    break;
#endif
#ifdef TARGA_SUPPORTED
  case FMT_TARGA:
    jselwtarga(cinfo);
    break;
#endif
  default:
    ERREXIT(cinfo->emethods, "Unsupported output file format");
    break;
  }
}


/*
 * Signal catcher to ensure that temporary files are removed before aborting.
 * Doesn't do anything during intervals where emethods is NULL.
 */

static external_methods_ptr emethods; /* for access to free_all */

LOCAL void
signal_catcher (int signum)
{
  if (emethods != NULL) {
    emethods->trace_level = 0; /* turn off trace output */
    (*emethods->free_all) ();   /* clean up memory allocation & temp files */
  }
  exit(EXIT_FAILURE);
}
```

```
LOCAL void
usage (void)
/* complain about bad command line */
{
  fprintf(stderr, "Usage: djpeg [switches] inputfile(s)\n");
  fprintf(stderr, "List of input files may use wildcards (* and ?)\n");
  fprintf(stderr, "If input file name has no extension, .JPG is assumed\n");
  fprintf(stderr, "Output filename is same as input filename, except for extension\n");
  fprintf(stderr, "Optional switches:\n");
#ifdef GIF_SUPPORTED
  fprintf(stderr, "  -G         Select GIF output, extension .GIF  (default)\n");
#endif
#ifdef PPM_SUPPORTED
  fprintf(stderr, "  -P         Select PPM/PGM output, extension .PPM\n");
#endif
#ifdef RLE_SUPPORTED
  fprintf(stderr, "  -R         Select Utah RLE output, extension .RLE\n");
#endif
#ifdef TARGA_SUPPORTED
  fprintf(stderr, "  -T         Select Targa output, extension .TGA\n");
#endif
  fprintf(stderr, "  -g         Force grayscale output\n");
  fprintf(stderr, "  -q colors  Quantize to no more than N colors\n");
  fprintf(stderr, "  -1         Use 1-pass quantization (fast, low quality)\n");
  fprintf(stderr, "  -D         Don't use dithering in quantization\n");
  fprintf(stderr, "  -b         Apply cross-block smoothing\n");
#ifndef FREE_MEM_ESTIMATE
  fprintf(stderr, "  -m memory  Maximum memory to use (default 300K); see
USAGE\n");
#endif
  fprintf(stderr, "  -d         Generate debug output\n");
  exit(EXIT_FAILURE);
}


/* Display progress report */
```

```
METHODDEF void
progress_monitor (decompress_info_ptr cinfo, long loopcounter, long looplimit)
{
 if (cinfo->total_passes > 1) {
  fprintf(stderr, "\rPass %d/%d: %3d%% ",
          cinfo->completed_passes+1, cinfo->total_passes,
          (int) (loopcounter*100L/looplimit));
 } else {
  fprintf(stderr, "\r %3d%% ",
          (int) (loopcounter*100L/looplimit));
 }
 fflush(stderr);
}


/*
 * Check for overwrite of an existing file; clear it with user
 */

LOCAL boolean
is_write_ok (char * outfilename)
{
 FILE * ofile;
 int ch;

 ofile = fopen(outfilename, READ_BINARY);
 if (ofile == NULL)
  return TRUE;                  /* not present */
 fclose(ofile);           /* oops, it is present */

 for (;;) {
  fprintf(stderr, "%s already exists, overwrite it? [y/n] ",
          outfilename);
  fflush(stderr);
  ch = getc(stdin);
  fflush(stdin);                /* flush rest of line */
  switch (ch) {
  case 'Y':
```

```
    case 'y':
      return TRUE;
    case 'N':
    case 'n':
      return FALSE;
    /* otherwise, ask again */
    }
  }
}


/*
 * Save current switch values here --- necessary for multiple input files!
 */

static boolean b_smooth;      /* -b */
static boolean gray_scale;    /* -g */
static int num_colors;        /* -q, or 0 if not seen */
static boolean two_pass_q;    /* not -1 */
static boolean do_dither;     /* not -D */
static long cur_mem;          /* -m, or value of FREE_MEM_ESTIMATE */
static int cur_trace;         /* -d */

#define MAX_FNAME_LEN 150


/*
 * Process a single input file
 */

LOCAL void
process_one_file (char * infilename)
{
  struct decompress_info_struct cinfo;
  struct decompress_methods_struct dc_methods;
  struct external_methods_struct e_methods;
  char outfilename[MAX_FNAME_LEN];
  int i;
```

```
/* Select the input and output files */

if ((cinfo.input_file = fopen(infilename, READ_BINARY)) == NULL) {
  fprintf(stderr, "djpeg: can't open %s\n", infilename);
  return;
}
/* Make outfilename be infilename with appropriate extension */
strcpy(outfilename, infilename);
for (i = strlen(outfilename)-1; i >= 0; i--) {
  switch (outfilename[i]) {
  case ':':
  case '/':
  case '\\':
    i = 0;                      /* stop scanning */
    break;
  case '.':
    outfilename[i] = '\0';      /* lop off existing extension */
    i = 0;                      /* stop scanning */
    break;
  default:
    break;                      /* keep scanning */
  }
}
switch (requested_fmt) {
case FMT_GIF:
  strcat(outfilename, ".GIF");
  break;
case FMT_PPM:
  strcat(outfilename, ".PPM");
  break;
case FMT_RLE:
  strcat(outfilename, ".RLE");
  break;
case FMT_TARGA:
  strcat(outfilename, ".TGA");
  break;
}
```

```
if (! is_write_ok(outfilename)) {
  fclose(cinfo.input_file);
  return;
}
if ((cinfo.output_file = fopen(outfilename, WRITE_BINARY)) == NULL) {
  fprintf(stderr, "djpeg: can't create %s\n", outfilename);
  fclose(cinfo.input_file);
  return;
}

/* Initialize the system-dependent method pointers. */
cinfo.methods = &dc_methods;
cinfo.emethods = &e_methods;
jselerror(&e_methods);        /* error/trace message routines */
jselmemmgr(&e_methods);  /* memory allocation routines */
dc_methods.d_ui_method_selection = d_ui_method_selection;

/* Now OK to enable signal catcher. */
emethods = &e_methods;

/* Set up JPEG parameters. */
j_d_defaults(&cinfo, TRUE);
cinfo.do_block_smoothing = b_smooth;
if (gray_scale)
  cinfo.out_color_space = CS_GRAYSCALE;
if (num_colors > 0) {
  cinfo.desired_number_of_colors = num_colors;
  cinfo.quantize_colors = TRUE;
}
cinfo.two_pass_quantize = two_pass_q;
cinfo.use_dithering = do_dither;
e_methods.max_memory_to_use = cur_mem;
e_methods.trace_level = cur_trace;

/* Start up progress display, unless trace output is on */
fprintf(stderr, "Decompressing %s => %s\n", infilename, outfilename);
if (cur_trace == 0)
```

```
  dc_methods.progress_monitor = progress_monitor;

  /* Set up to read a JFIF or baseline-JPEG file. */
  /* A smarter UI would inspect the first few bytes of the input file */
  /* to determine its type. */
#ifdef JFIF_SUPPORTED
  jselrjfif(&cinfo);
#else
  You shoulda defined JFIF_SUPPORTED.  /* deliberate syntax error */
#endif

  /* Do it to it! */
  jpeg_decompress(&cinfo);

  /* Clear away progress display */
  if (cur_trace == 0)
    fprintf(stderr, "\r                \r");
  fflush(stderr);

  /* All done. Close files, disable signal catcher */
  fclose(cinfo.input_file);
  fclose(cinfo.output_file);
  emethods = NULL;
}


/*
 * Process one filespec; expand wildcards
 */

LOCAL void
process_filespec (char * filespec)
{
  FF_DATA ffblock;
  char infilename[MAX_FNAME_LEN];
  int pathlen, i;
  boolean hasext;
```

```
/* Determine whether there is a path and an extension */
pathlen = 0;
hasext = FALSE;
for (i = strlen(filespec)-1; i >= 0; i--) {
  switch (filespec[i]) {
  case ':':
  case '/':
  case '\\':
    pathlen = i+1;              /* path part ends here */
    i = 0;                      /* stop scanning */
    break;
  case '.':
    hasext = TRUE;              /* there is an extension */
    break;
  default:
    break;                      /* keep scanning */
  }
}

/* Make infilename be filespec; if no explicit extension, attach .JPG */
strcpy(infilename, filespec);
if (! hasext)
  strcat(infilename, ".JPG");

/* Scan over matching files */
if (FINDFIRST(infilename, &ffblock) != 0) {
  fprintf(stderr, "No files matching %s\n", infilename);
} else {
  do {
    strcpy(infilename, filespec);
    strcpy(infilename + pathlen, ffblock.FF_NAME);
    process_one_file(infilename);
  } while (FINDNEXT(&ffblock) == 0);
 }
}


/*
```

```
 * The main program.
 */

GLOBAL int
main (int argc, char **argv)
{
  int c;

  /* Set up signal catcher. */
  emethods = NULL;
  signal(SIGINT, signal_catcher);
#ifdef SIGTERM                  /* not all systems have SIGTERM */
  signal(SIGTERM, signal_catcher);
#endif

  /* Scan command line, process switches */

  requested_fmt = DEFAULT_FMT; /* initialize default switch values */
  b_smooth = FALSE;
  gray_scale = FALSE;
  num_colors = 0;
  two_pass_q = TRUE;
  do_dither = TRUE;
#ifdef FREE_MEM_ESTIMATE
  cur_mem = FREE_MEM_ESTIMATE;
#else
  cur_mem = 300000L;            /* reasonable default for DOS */
#endif
  cur_trace = 0;

  while ((c = egetopt(argc, argv, "GPRTbgq:1Dm:d")) != EOF)
    switch (c) {
    case 'G':                   /* GIF output format. */
      requested_fmt = FMT_GIF;
      break;
    case 'P':                   /* PPM output format. */
      requested_fmt = FMT_PPM;
      break;
```

```
case 'R':                       /* RLE output format. */
  requested_fmt = FMT_RLE;
  break;
case 'T':                       /* Targa output format. */
  requested_fmt = FMT_TARGA;
  break;
case 'b':                       /* Enable cross-block smoothing. */
  b_smooth = TRUE;
  break;
case 'g':                       /* Force grayscale output. */
  gray_scale = TRUE;
  break;
case 'q':                       /* Do color quantization. */
  if (optarg == NULL)
      usage();
  if (sscanf(optarg, "%d", &num_colors) != 1)
      usage();
  break;
case '1':                       /* Use fast one-pass quantization. */
  two_pass_q = FALSE;
  break;
case 'D':                       /* Suppress dithering in color quantization. */
  do_dither = FALSE;
  break;
case 'm':                       /* Maximum memory in Kb (or Mb with 'm'). */
  { long lval;
      char ch = 'x';

      if (optarg == NULL)
        usage();
      if (sscanf(optarg, "%ld%c", &lval, &ch) < 1)
        usage();
      if (ch == 'm' || ch == 'M')
        lval *= 1000L;
      cur_mem = lval * 1000L;
  }
  break;
case 'd':                       /* Debugging. */
```

```
    /* On first -d, print version identification */
    if (cur_trace == 0)
        fprintf(stderr, "Independent JPEG Group's DJPEG, version %s\n%s\n",
                JVERSION, JCOPYRIGHT);
    cur_trace++;
    break;
  case '?':
  default:
    usage();
    break;
  }

 /* Process file specifications */
 if (optind >= argc)
  usage();                      /* no filespecs?? */

 while (optind < argc)
  process_filespec(argv[optind++]);

 /* All done. */
 exit(EXIT_SUCCESS);
 return 0;                      /* suppress no-return-value warnings */
}
```