Automation of the Electron Paramagnetic Resonance Spectrometer

Shalini Verghese

A Thesis

in

The Department

of

Physics

Presented in Partial Fulfillment of the Requirments
for the Degree of Master of Science at
Concordia University
Montreal, Quebec, Canada

August 1989

ISBN 0-315-51366-7

ABSTRACT

Automation of the EPR Spectrometer

Shalini Verghese

The Electron Paramagnetic Resonance Spectrometer was connected to a computer to facilitate the data acquisition and storage. The SWTPC MC6809 microcomputer was used along with a digitizer from Tracor Northern to acquire data. Software in low level language, for the MC6809 microprocessor, was written to receive data from the Tracor Northern digitizer and store it in files on a floppy disk. Due to hardware difficulties the data acquisition system was changed to an IBM data acquisition card and an IBM compatible microcomputer. The software (eprware) used for data acquisition and manipulation for the IBM data acquisition system has been described in this work.

Intensity variation with temperature for $CdCl_2 2NiCl_2 . 12H_2O$, icosahedral $Al_{55}Si_{25}Mn_{50}$ and amorphous $Al_{55}Si_{25}Mn_{50}$ were observed. The critical temperature T_c and the critical parameter γ' were calculated for the latter two samples. The principle g-values of the high T_c superconductor $YBa_{1.9}Na_{0.1}Cu_3O_{7-\delta}$ were also calculated.

ACKNOWLEDGEMENTS

The author wishes to express deep gratitude to Dr. S.K. Misra and Dr. N. Eddy for proposing this study and for the interest, patience and invaluable assistance expressed by them.

The author is thankful to Dr. L.E. Misiak, Mr. A. Christodoulopoulos, others in the Physics workshop and the Department of Physics for assistance provided.

The author especially appreciates the encouragement and support shown by family and friends.

Grateful acknowledgement is also made of financial support from Dr. S.K. Misra's N. S. E. R. C. grant (No. A4485).

TABLE OF CONTENTS

LIST OF FIGURES AND TABLES

CHAPTER I

INTRODUCTION

Paramagnetism is caused by the magnetic moment of an unpaired electron. Substances exhibit paramagnetism when the molecules or atoms have an unpaired electron, which gives rise to a resultant angular momentum. When one considers this angular momentum, thus the magnetic moment, attributed to electrons it is called electron paramagnetism. This results from net spin, or net orbital angular momentum of electrons, or from their combination. Thus we see paramagnetism in free radicals, free atoms, ions, etc.

Electron Paramagnetic Resonance (EPR) is the resonance of electronic magnetic dipole moments. The unpaired electron in a paramagnetic substance makes transitions between two energy states when the substance is exposed to microwaves whose quantum of energy equals the energy difference between the levels participating in resonance. For X-band EPR, the correct amount of energy is given by electromagnetic waves of frequency in the range of 9.5 to 10 GHz, known as microwaves. Therefore microwaves are required to study EPR experimentally.

1.1 Spectrometer arrangement

The experimental setup for the detection of magnetic dipolar transitions in paramagnetic substances is known as the EPR spectrometer. The spectrometer arrangement consists of a cavity system, a microwave source, the magnet system, and the modulation and detection systems (see fig.1.1) [1].

1.1.1 The cavity system

The cavity is the focus of an EPR spectrometer. Since in X-band EPR, microwaves are used, whose wavelength is of the order of centimeters, the cavity dimensions are conveniently large. To be useful for electron spin resonance the cavity mode should permit (i) a high microwave energy density, (ii) the sample to be placed at a position where the magnetic field is maximum, and (iii) the pependicularity of the static external Zeeman field and the magnetic component of the microwave field.

The sharpness of response of any resonant system is described by a factor of merit, Q, which is defined for a resonant cavity as follows:

$$Q = 2\pi \frac{(\text{ maximum microwave energy stored in the cavity})}{\text{energy dissipated per cycle}}$$

Microwave energy is allowed into and out of the sample cavity by a small hole, known as an "iris".

Standing waves arise in the cavity due to the reflection of some of the incident microwave energy. Power at the detector should be only from reflections originating at the microwave cavity. The slide-screw tuner allows one to set up standing waves of such amplitude and phase as to minimize the existing standing waves. This is accomplished by varying the depth of insertion and the position of a

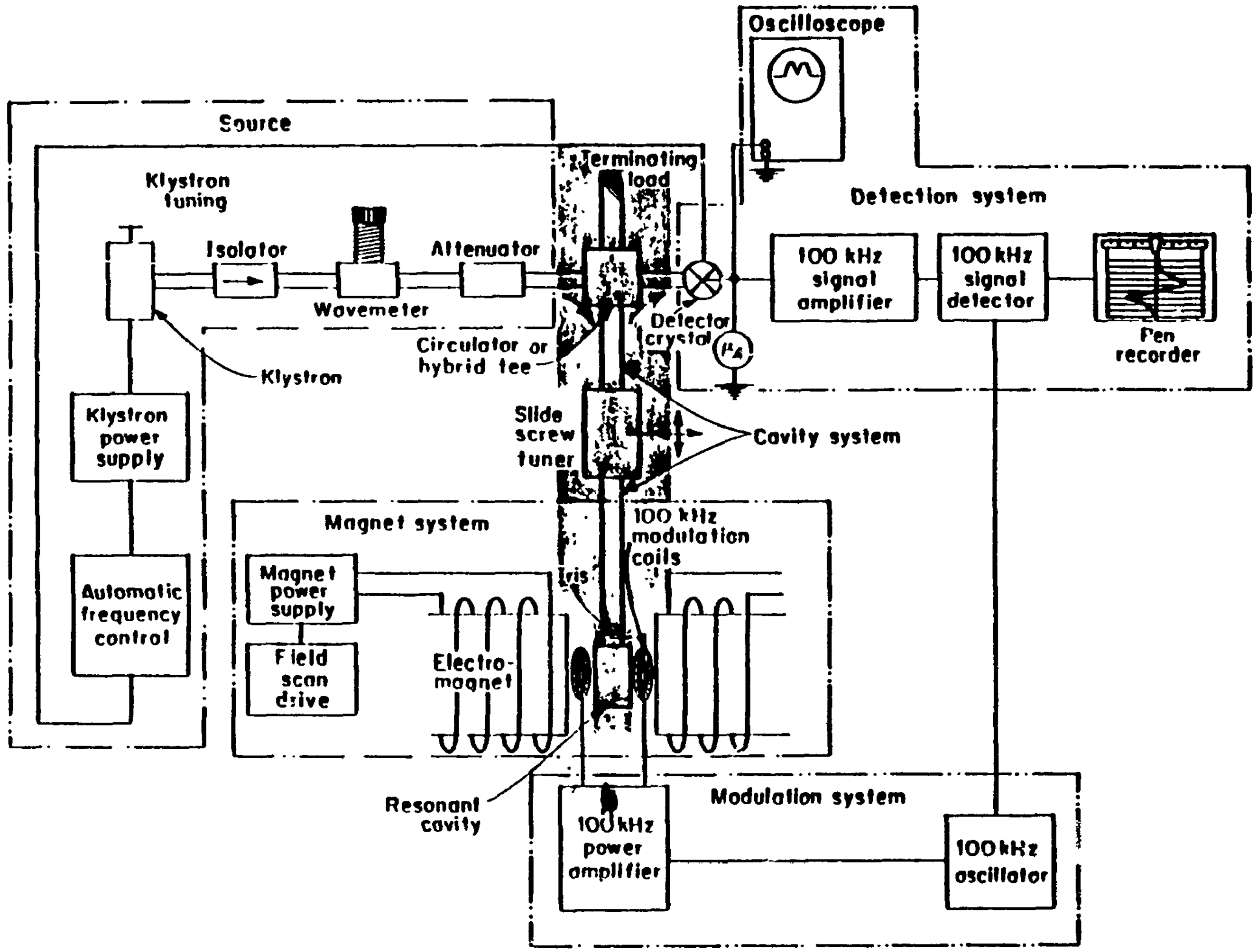


FIG. 1.1 BLOCK DIAGRAM OF AN X-BAND ESR SPECTROMETER

small metallic probe along the waveguide.

A circulator directs microwave power to the cavity and reflects the signal from the cavity to the detector.

1.1.2 The source of microwaves, isolator, wavemeter and attenuator

The klystron is the source of microwave radiation. The isolator passes microwave energy in the forward direction, while it strongly attenuates any reflection. In this way it minimizes variation in the klystron frequency due to backward reflectors.

The wavemeter is a cylindrical resonant cavity. The length of which is adjustable by means of a micrometer.

The attenuator adjusts the level of the microwave power incident on the sample.

1.1.3 The magnet system

The magnet is the source of static magnetic field. The field should be stable and uniform over a given volume. The field controller and the power supply allow the setting of the range of the magnetic field, sweep time and other parameters that are required to be changed for a particular setup.

1.1.4 The modulation and detection system

The main disadvantage of the above experimental set up is the contribution of noise to the signal. This is reduced by using a small amplitude modulation of the magnetic field which cuts out noise-contributing components to frequencies very close to the modulation frequency. The modulation system consists of small Helmholtz coils on each side of the cavity along the axis of the static field.

The detection system mixes the signal with the output of a local oscillator to produce an intermediate frequency which is amplified and detected. In this case the microwave signal that is reflected from the cavity is mixed with the output of a local oscillator. This frequency is usually 30 MHz above or below that of the signal Klystron. This is known as superheterodyne detection.

1.2 The local EPR system

The Concordia University Physics Department uses the Varian V-4502 EPR spectrometer system, and a Bruker field controller of the magnetic field and a Bruker magnet power supply. The recorder of the signal is a Graphtec X-Y plotter. The magnet is a twelve-inch Varian Associates magnet (model no. V-3900). It is the source of the external Zeeman magnetic field.

The EPR cavity, in which the sample is placed, is a Varian Associates (V-4531) multipurpose cavity; so called because it can be used for fixed and variable temperature studies. It is designed for 100 KHz field modulation, and provides a 35 G peak-to-peak field modulation at this frequency.

The klystron is supplied with resonator voltage by a Varian Associates (V-4500-20) klystron power supply. The output, as a function of frequency, is called the klystron mode. A klystron operates in several modes but the experimentalist usually selects the mode corresponding to the highest power output. The mode can be displayed on the oscilloscope of the V-4595 selector panel. The controls are also present on the selector panel.

The Varian V-4560 EPR spectrometer utilizes the 100 KHz field modulation and control unit. The detector noise varies inversely with the frequency of modulation. Therefore, the higher the frequency of the field modulation the better the EPR signal-to-noise ratio. 100 KHz is found to be an optimum choice, taking into account field modulation, penetration of cavity walls, sample relaxation times and crystal noise [2].

The field control and measuring unit is a Bruker B-H15 unit. It consists of a built-in microprocessor system to make it accurate. The field controller can function in different modes. It can be put in remote and be controlled externally or perform independently. It can be programmed for a specific field sweep and sweep time. The duration for which the pulse from the address advance remains low varies with the sweep time. The address advance is a BNC female connector on the rear panel of the field controller. The output is a positive going pulse that changes the sweep address by one. This pulse is required for data acquisition as described later in Chapter IV.

The recorder is a GRAPHTEC X-Y recorder. The EPR signal is sent across the positive and negative terminal of the Y axis, and the time sweep is sent to the positive and negative terminal of the X axis.

With this spectrometer setup, the EPR experiments can be done at room, liquid-nitrogen or liquid-helium temperatures. The scan time, the maximum and minimum field and step size etc, can be programmed, using the field controller. The spectrum is then used to calculate the required parameters.

Many experiments in physics are aided by microprocessors for data acquisition and analysis. The EPR spectrometer system at Concordia is not computerized. The object of the project covered in the present thesis is to manipulate the magnetic field controller and to acquire the EPR data using a microcomputer (see figure 1.2, details of connections in Chapter IV section 4.5). The EPR data is currently obtained as an analog signal on the GRAPHTEC plotter. In order to use the microcomputer the EPR signal has to be digitized. With the data in digital form, appropriate software can be used to analyse the data. As for measuring the magnetic field, which is represented on the X-axis, the digitization has already been performed, since the Bruker field controller already has a built-in microprocessor. The spectra of a few samples should be acquired and analyzed to demonstrate the automation of the

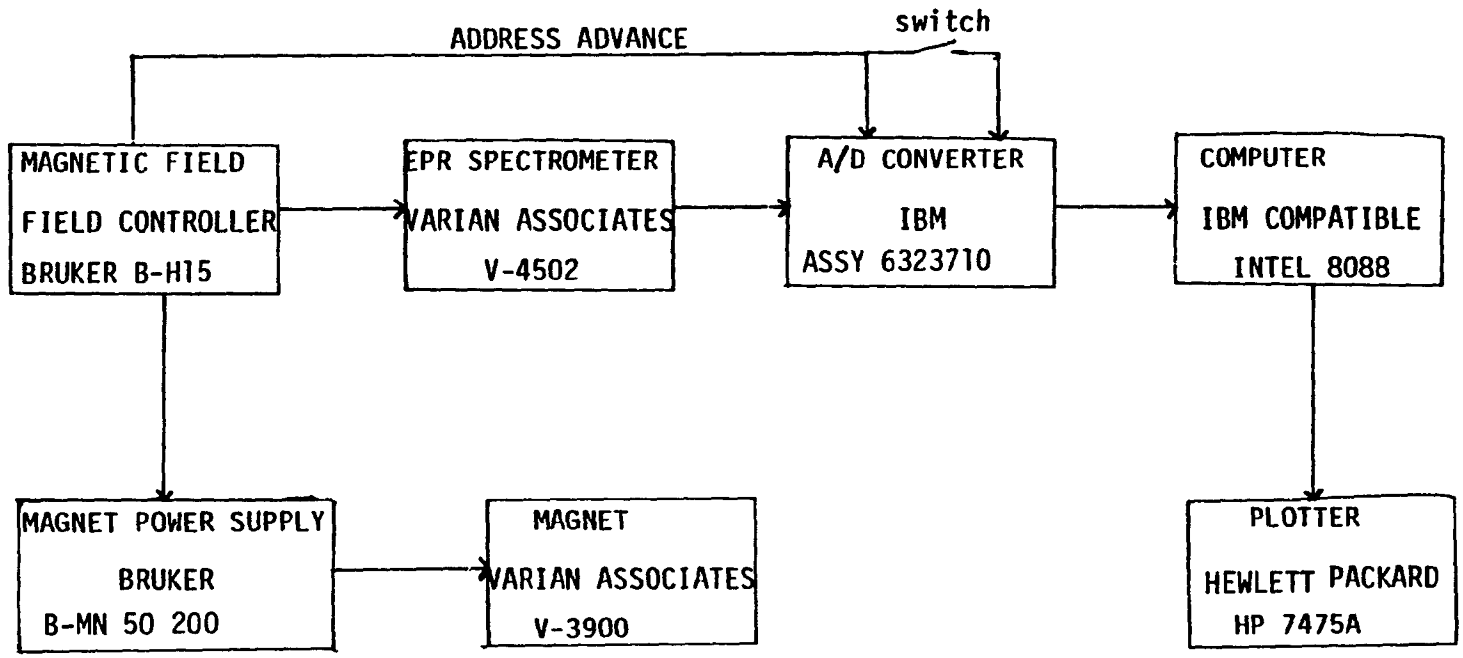


FIG. 1.2 BLOCK DIAGRAM OF AUTOMATED EPR SPECTROMETER

available equipment.

To this end, the spectrum of a sample of $YBa_{1.9}Na_{0.1}Cu_3O_{7-\delta}$ was taken and curve-fitted to find the individual peaks from which the principle g_1, g_2, and g_3 values were calculated. Also spectra for samples of icosahedral $Al_{55}Si_{25}Mn_{20}$, amorphous $Al_{55}Si_{25}Mn_{20}$ and $CdCl_2.2NiCl_2.12H_2O$ were obtained at different temperatures. From this the intensity was calculated for all samples and its relation to temperature was noted. Particular interest was paid to the temperature range around critical temperature for the $Al_{55}Si_{25}Mn_{20}$ samples. The critical temperature T_c and the critical parameter γ' were calculated for the two samples of $Al_{55}Si_{25}Mn_{20}$.

The thesis is divided into two parts, work that was done on the Southwest Technical Products microcomputer (MC6809 microprocessor) and that which was done on an IBM compatible microcomputer (Intel 8088 microprocessor). The SWTP computer was intended to be interfaced to a Tracor Northern (TN) digitizer which in turn was to be connected to the EPR spectrometer. Programs were written to collect data from a serial port, write data files onto a floppy disk and print data. All the software was written in MC6809 assembly language. However, a catastrophic electrical failure of the TN put an end to this project. The cost of repair was found to be more than replacing the existing system with a more recent system. So a new system consisting of an IBM analog-to-digital conversion card

(ADC) plugged into an IBM compatible microcomputer was obtained. The software to acquire and display the data (EPRWARE) was obtained from Dr. Philip Morse of the University of Illinois, Chicago. Chapter II deals with the EPR spectra and details connected to this project. Chapter III explains the operating system in general and gives a brief but related account of the couple of operating systems used. Chapter IV deals with some of the hardware that was needed for the acquisition of the data and a few of the changes that had to be made. It also explains some of the software used. Chapter V gives details and results of the spectra analyzed. Chapter VI, the concluding chapter, is a summary of what was done and a few suggestions for future work.

CHAPTER II

ACQUISITION OF AN EPR SPECTRUM AND MANIPULATION OF DATA

This chapter deals with the EPR spectrum and the useful parameters that can be calculated from it.

EPR spectrometers usually record the data in the form of the first derivative of the EPR absorbtion line. The EPR signal represents a very small change in the overall absorbtion of microwave power in the cavity. This requires that the signal be amplified, which is done by adding a small oscillating modulating magnetic field to the large external field. If $\vec{H}_0$ is the the external large field and $\vec{H}_m$ is the modulation field then the resultant $(\vec{H}_o + \vec{H}_m)$ swings back and forth, alternately, increasing and decreasing the absorption height. Accordingly, the dc (direct current) output alternately increases and decreases. This increase and decrease of the dc provides an alternating current to be amplified and recorded.

The EPR signal allows us to evaluate various terms in the spin Hamiltonian, relative intensity etc. Phase-transition mechanisms can also be studied using EPR. In the following paragraphs a little will be said about intensity and g factors.

The g factor appears in the magnetogyric ratio (γ) as a factor that is required if there is spin angular momentum and orbital angular momentum. For free electron spin the value of g is very close to 2. The g factor is a function of the resonant magnetic field value, which in turn is a function of the field orientation relative to the crystal axes. If the principle magnetic axes of the molecule are labeled X, Y, Z, then g_x is the g factor for H along the X-axis, g_y is that for H along the Y-axis and g_z is that for H along the Z-axis. In a powder sample all the g factors can be determined from one spectrum, since the powder spectrum does not change upon changing the orientation of the external magnetic field. The g factors from the powder spectrum were estimated for the sample of $YBa_{1.9}Na_{0.1}Cu_3O_{7-\delta}$. (Details are given in chapter V.)

The intensity of an EPR line can be calculated by integrating the full absorption curve. If the curve is the first derivative, two consecutive integrations are to be done. On the other hand, the approximate relative intensity $\mathcal{I}$ of a line can be obtained for the first-derivative signal by :

$$\mathcal{I} \alpha Y_{max}(\Delta H_{pp})^2$$

$2Y_{max}$ is the peak-to-peak first-derivative amplitude and ΔH_{pp} is the peak-to-peak width. With the computer the data can be collected and stored in files. The data then can be numerically integrated to give the area under the specified portion of the curve to yield the intensity. The variation of intensity with temperature for the samples $CdCl_2.2NiCl_2.12H_2O$ and $Al_{55}Si_{25}Mn_{20}$ were obtained in this manner.

The relationship between susceptibility (χ) and critical temperature (T_c) is given by;

$\chi=(T_c-T)^{-\gamma'}$ for $T < T_c$.

γ' is the critical parameter when $T < T_c$. Susceptibility is proportional to intensity $\mathcal{I}$ therefore $\mathcal{I}=(T_c-T)^{-\gamma'}$. Using this relation the critical temperature and the critical parameter were calculated for a sample of amorphous $Al_{55}Si_{25}Mn_{20}$ and icosahedral $Al_{55}Si_{25}Mn_{20}$.

CHAPTER III

OPERATING SYSTEMS IN MICROCOMPUTERS

When programming in low level languages it is essential to know the routines used by the operating system. In this chapter two operating systems, Flex and Microsoft disc operating system, are discussed briefly. Instructions to input and output data for both systems are looked into. Since speed is an important factor that has to be considered when writing programs to collect data, it should be done in assembly language (a low level language). Programs in assembly require less time to run.

An operating system (OS) is an organized collection of programs that acts as an interface between machine hardware and users. It provides users with facilities to simplify the design, coding, debugging and maintenance of programs. It also controls the allocation of resources to assure efficient operation.

The major constituent of an operating system can be divided into two overlapping classes. First are those which may be called directly, or indirectly, during user program execution; i.e, the program uses commands that require these routines: Input/ Output routines (I/O) and and other file services are a few examples. The second class consists of those programs that are invoked explicitly through control language statements issued by users.

The interface between the operating system and the user programs is a set of extended instructions that the operating system provides, called system calls, e.g, reading and writing files.

Operating systems store information in files. One category of system calls relates to the file system. They are needed to create, remove, read and write files.

When a file is open it can be read from or written into. If a read instruction is intended, the programmer must verify that the file is actually opened for read. Once verified some variables must be initialized. The read operation is actually a loop which sorts the data into blocks, each of which fits a single disk sector. A block begins at the current position and extends until one of the following conditions is met;

a) all bytes have been read from buffer

b) a block boundary is encountered

c) the end-of-file is encountered

The program requires that a block of the file is loaded into a buffer and the buffer address specifed, as well as the number of bytes that must be read. A counter is used to keep track of the number of bytes read. The counters and pointers to the buffers are updated after a block of data is read. A similar procedure is followed for write, except here space has to be clear to write the file on disk or other storage devices.

3.1 Flex operating system

The Flex operating system has versions for the 6800 and

6809 microprocessors. It consists of a disk operating system (DOS) which processes commands, the file management system (FMS) manages files and a set of user callable commands.

The DOS helps the user to utilize all commands to communicate with the FMS. The user has to use a code to call the FMS as a subroutine. It is useful in that a programmer doesn't always have to write low-level file management routines every time a program is written. The DOS also removes hardware interfacing from the application program, like initializing registers.

FMS is the software that is the link between the DOS and the actual disk hardware. The information about a file is in file control blocks (FCB). It might help if a little is said about the role of FCB in Flex. The FCB is actually a 320 byte block of random access memory (RAM) in the user program area. RAM is the region in memory that can be written into and read from. The user can specify any location in the user program area as a FCB for a specific file. Each file being used simultaneously will have to have its own FCB. The FMS is accessed by commands in the form of codes. Each code does a specific function. For example, if a file needs to be opened to be read, in Flex one would issue the following commands:

LDX #FCB	load the x register with the file control block address.
LDA #1	load the accumulator with the specific

code to open for read, namely one (1).

STA 0,X this stores the code in the first byte of the file control block

JSR FMS call FMS

In Flex the file structure is linked. A linked structure allows the file to be fragmented and stored anywhere without user intervention. The FMS does the allocation of the file in the disk space. Even though the file is fragmented the logical sequence is not lost. The advantage of this is that if after the first few files are written on the disk, the user deletes the first one and decides to write a longer one, the remaining files need not be moved. The file is broken up so that it fits in the vacated spaces located elsewhere. A disadvantage is that it takes longer to read a file since the head has to locate all the different fragments on the disk.

The above mentioned are a few of the tools one needs to be familiar with for external communication, storage and transmission of data.

The software for external communication does basically two things, it acquires data from the interface (connects the external device to the CPU) and places it at the the required location as specified by the software so that it is available to the user. The program PS text (listed in the Appendix) sends data that is stored on the disk byte by byte to an external device like a printer or another monitor. The first operation involved is to acquire the file that has been specified. This is done by a DOS command called GETFIL. Whatever is typed in after the DOS command, is taken as the name of a file if the program requires it, and is put in a buffer (a block of reserved bytes of memory). GETFIL then uses the information to make the file available. The file is then opened to 'Read', a function done by the FMS. When a file is stored on the disk the FMS divides the file into sections and stores it in the free sectors. It also keeps a record of the tracks and sectors.

The starting disk address of the file is stored in the first 17-18 bytes of the FCB. The bytes 64-68 contain the track and sector of where the remaining file is stored. Once the file is found and a sector is read into the buffer then the contents of the buffer can be sent out to any external device through an interface. A similar procedure is used to get data from an external device and save on the disk except this time the FMS uses the 'Write' command.

3.2 Microsoft disk operating system

Microsoft disk operating system (MSDOS) is a very popular disk operating system for personal computers based on Intel's 8086 and 8088 microprocessers. MSDOS allows communication with compatible devices through the RS-232-C serial interface.

The Intel 8088 has four 16-bit registers ax, bx, cx, and dx. Within each are two 8-bit registers ah, al, bh, bl etc, where h and l are high order and low order bytes respectively. In MS DOS most I/O is accomplished via software interrupt #21h the hexadecimal code. A function number has to be placed in the ah register, and the relevant information in the other registers as required by the specific service call.

When MSDOS takes control it switches to an internal stack. The user registers except ax are preserved unless the request indicates otherwise.

Functions of the MSDOS are device input/output and file access. Our primary interest is in auxiliary input and output since this is what is involved in data collection from an external device. To get a character from an auxiliary device (function 3) the number 3 is put in the register ah and the register al has the return value which is an ASCII character. To output a character to an auxiliary device (function 4) one has to load register ah with 4 and the character to be sent out in dl.

Each file being accessed through the operating system must have a corresponding FCB. The FCB data area consists of a sequence of 33 bytes for sequential access and a series of 37 bytes when the file is accessed randomly.

Flex has seperate function codes for files to be opened for write and for those to be opened for read. Unlike Flex, MS DOS has a function code to open the file after which depending on whether a read or write is required the appropriate code is used.

CHAPTER IV

THE HARDWARE AND SOFTWARE FOR DATA ACQUISITION AND ANALYSIS

In this chapter the hardware and software that have been used for data acquisition will be described briefly. Two systems were used. First the MC6809 South West Tech Personal computer (SWTPC) with the Tracor Northern digitizer were used and then later the IBM-PC compatible with the IBM data acquisition card were utilized.

4.1 The SWTPC 6809 computer system

The SWTPC uses a MC6809 microprocessor, which is an 8-bit microprocessor in a 40 pin configuration. It has two 8-bit accumulators A and B, which can be manipulated and used as a 16-bit unit called D, with A as the most significant byte. There are two 16-bit index registers (X,Y), one 16-bit user stack pointer (U), one 16-bit hardware stack pointer (S), one 16-bit program counter (PC), one 8-bit direct page register (DP) and one 8-bit condition code register (CC). The microprocessor can handle 64K of memory.

4.2 FD-2 Controller board

The FD-2 controller board can control up to 4 disk drives, i.e it can send out the required prompts for the working of four disc drives, single density to double density. It is compatible with the MC6809 and MC6800 microprocessors. By repositioning jumpers (small connectors used to bridge gaps on a circuit board) the board can be configured for either a MC6800 or a MC6809 microprocessor. This FD-2 board is configured for the MC6809 microprocessor and for sixteen addresses per I/O slot (see fig 4.1). The upper left top is the decoding circuitry and the outputs on the right side accesses the drives. The WD2797 is the floppy disk formatter/controller. The numbered lines at the bottom left of the figure 4.1 are the ones altered for the MC6809 since there was a difference in the numbering on the board and on the circuit diagram of the controller board. The original figure is for the MC6800 [3]. It is currently used to drive two 5 1/4 inch disk drives connected to a SWTPC 6809. The cables and casing for the drives were made in the Science Technical Service Centre at Concordia University.

The FD-2 board was installed in port 1 (a slot at the rear of the computer which is allocated a certain address) of the SWTPC microcomputer. The command used to access the drives is the 'U' command in the SWTPC SBUG-E 6809 monitor program resident in the read only memory (ROM).

There was a problem encountered during installation. Both drives were accessed at the same time. When drive 0 was accessed nothing happened, but when drive 1 was accessed both drives worked. Systematically checking the jumpers on the FD-2 board showed the problem to be elsewhere. All the correct signals were transmitted at the

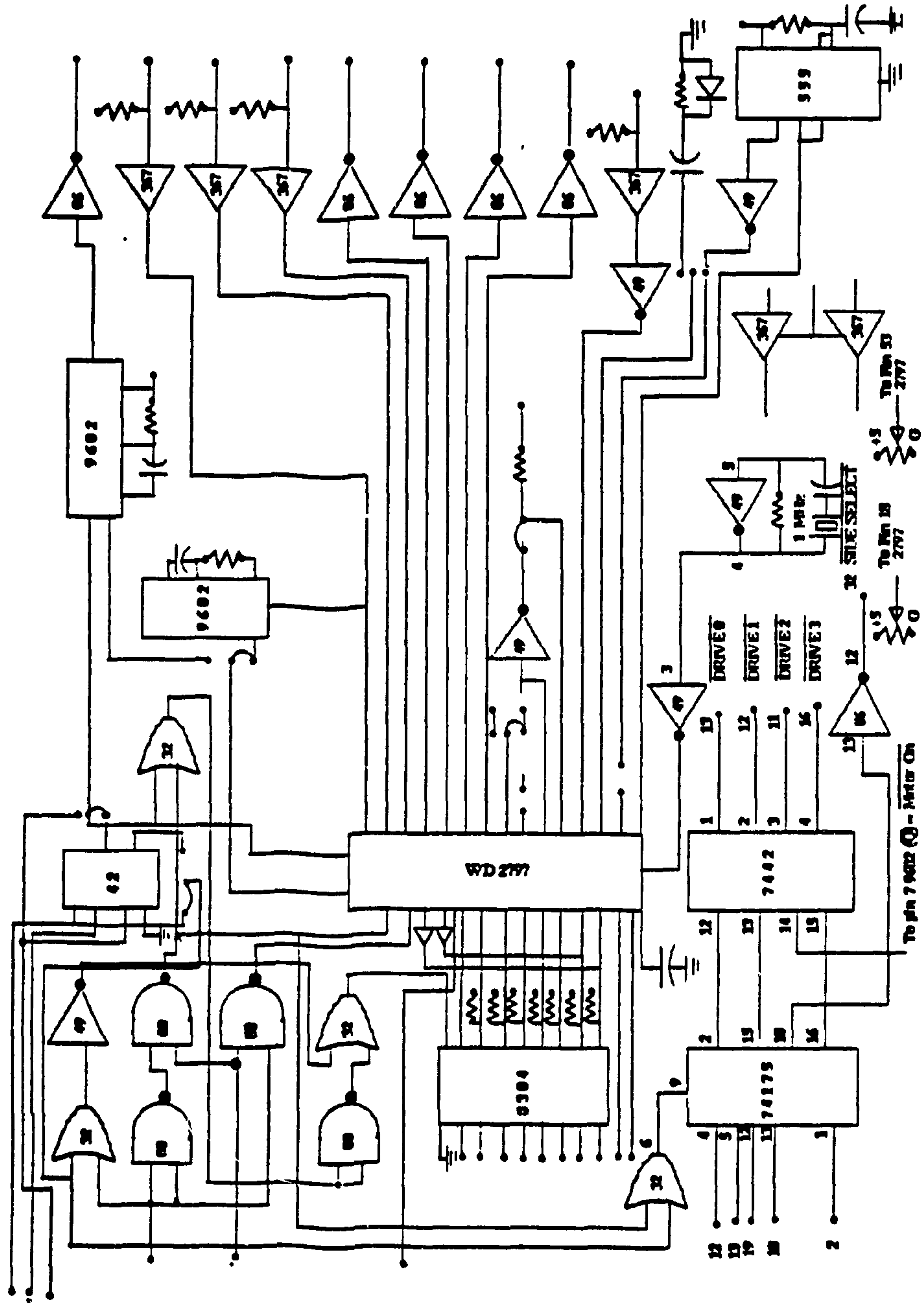


FIG. 4.1 FD-2 CONTROLLER BOARD

selection of a particular drive. This pointed the problem to the drives themselves and not the FD-2 board. The problem was discovered as soon as the covers were taken off the drives. Both drives were connected to act as drive 1. The output from the 7442 chip (see fig.4.1) for drive 1 was connected to access drive 0 and 1. After the minor adjustment of the connection was made the disk drives functioned normally.

4.3 64K Dynamic memory board (model DRAM-64)

The 64K dynamic memory board obtained from CPI, Hixson, Tennessee can be configured to have a memory capacity of 16K, 32K, 48K, or 64K bytes using 4116 memory chips. It has a refresh cycle at a frequency of 1 MHz. It operates with both the 6800 and 6809 systems. The memory capacity had to be increased because the Flex operating system requires 8K of memory starting from the address location hex C000 to DFFF and to run programs for collecting data.

The necessary components had to be purchased and soldered into place. All the required integrated circuits were obtained easily except the delay line (2 pieces 0447 -0250 -01) and two MSI circuits (MC3480, MC3242) which were discontinued by Motorola. The MSI circuits were kindly donated by Frank West of Georgia the delay lines were finally obtained from Tenessee and were the last two remaining of their kind. The 4116 200NS memory chips were taken off old boards donated by the Computer Science Department of Concordia University.

Using an oscilloscope all the voltage levels at the pins of the memory chips were determined to be correct. A portion of the circuit diagram of the board is shown in figure 4.2 [4]. After considerable effort, it was found that the 74LS245 (Octal bus trancivers with 3-state outputs) was not being enabled. The 74LS245 is enabled only if pin 19 (see fig. 4.2) goes low; i.e it has a voltage less than 2.5 volts. Tracing the enable pin 19 (the pin that makes the chip function) back to pin 3 of the 74LS00 (quadruple 2-input positive nand gates) it was found that pin 3 never went low. For pin 3, which is the output of the 74LS00 to be low the inputs of the gate should be high (see fig. 4.3). Checking the inputs to the 74LS00 showed that the input coming from pin 9 of the 74LS133 (13-input positive nand gates) to the 74LS00 was always low. For the output of the 74LS133 to be high one or more inputs should be low (see fig. 4.4). This meant checking the inputs which are the outputs of the 74LS138 (3-line to 8-line decoders/demultipliers). These were always high. The outputs of the 74LS138 will be high only if either of the enable inputs G1 (pin 6) is low or when G2 is high irrespective of what the other inputs are (see fig. 4.5).

The trouble was located at pin 4 of the 74LS138 which was not grounded properly and therefore was a high. Once

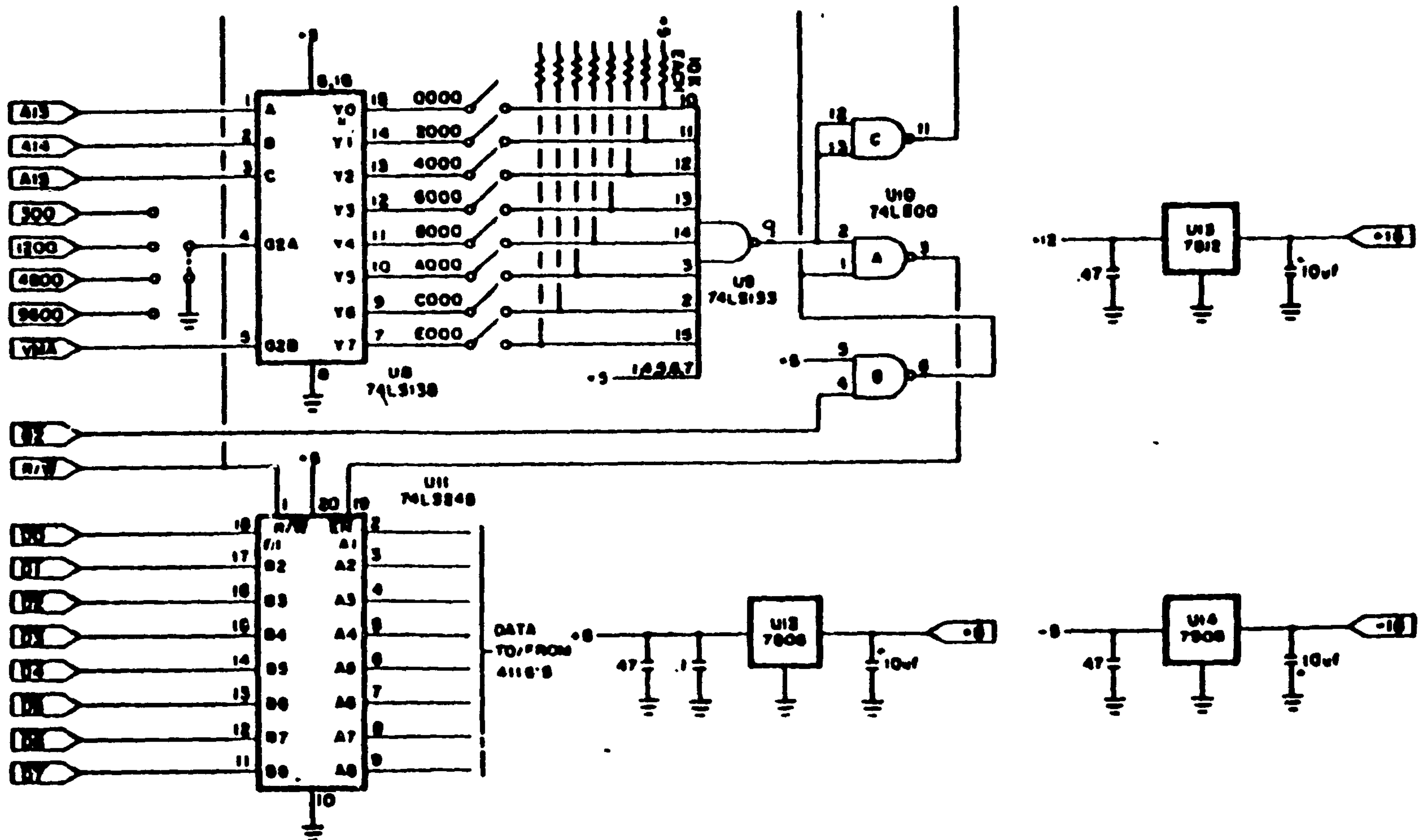


FIG. 4.2 64K DYNAMIC RAM BOARD

FUNCTION TABLE

INPUTS		OUTPUT
A	B	Y
H	H	L
L	X	H
X	L	H

LOGIC DIAGRAM

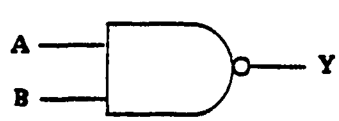


FIG. 4.3 74LS00 QUADRUPLE 2–INPUT POSITIVE NAND GATES

FUNCTION TABLE

INPUTS A THROUGH M	OUTPUT Y
ALL INPUTS H	L
ONE OR MORE INPUTS L	H

LOGIC DIAGRAM

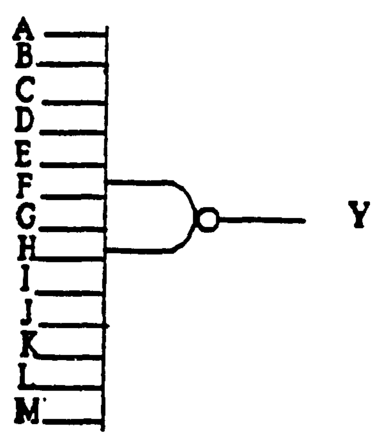


FIG. 4.4 74LS133 13–INPUT POSITIVE NAND GATES

FUNCTION TABLE

INPUTS					OUTPUTS							
ENABLE		SELECT										
G1	G2*	C	B	A	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
X	H	X	X	X	H	H	H	H	H	H	H	H
L	X	X	X	X	H	H	H	H	H	H	H	H
H	L	L	L	L	L	H	H	H	H	H	H	H
H	L	L	L	H	H	L	H	H	H	H	H	H
H	L	L	H	L	H	H	L	H	H	H	H	H
H	L	L	H	H	H	H	H	L	H	H	H	H
H	L	H	L	L	H	H	H	H	L	H	H	H
H	L	H	L	H	H	H	H	H	H	L	H	H
H	L	H	H	L	H	H	H	H	H	H	L	H
H	L	H	H	H	H	H	H	H	H	H	H	L

*G2 = G2A + G2B

H = high level, L = low level, X = irrelevant

LOGIC DIAGRAM

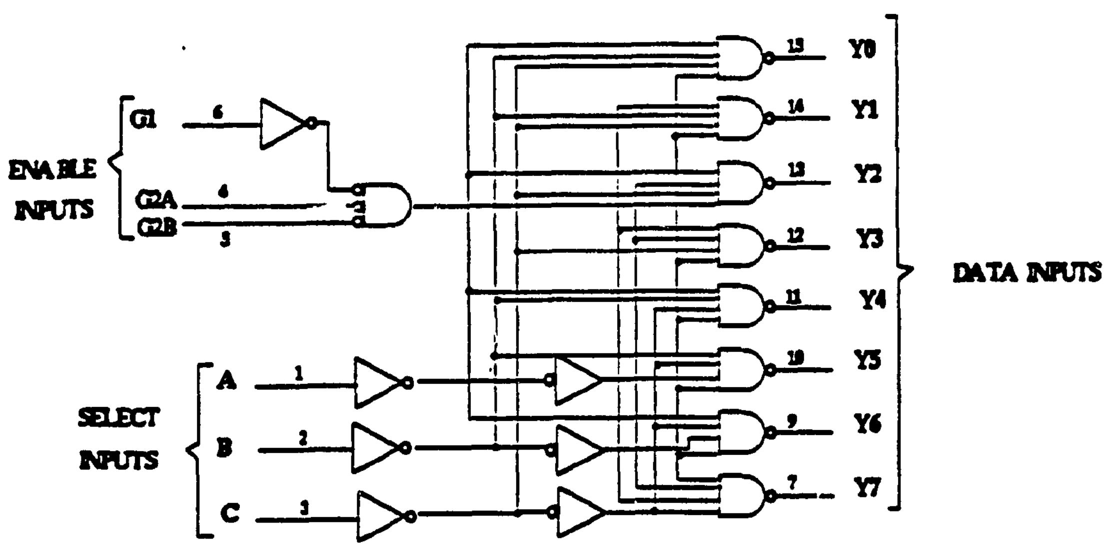


FIG. 4.5 74LS138 3-LINE TO 8-LINE DECODERS/ DEMULTIPLEXERS

this was grounded the 74LS138 was enabled but still the board did not function.

The delay line could not be checked, since the specifications were not available. Likewise, the data sheets for the MC3480 (dynamic memory controller) and MC3242A (address multiplexer and refresh address counter) were unobtainable. These chips could be the cause of the board not functioning, or any one of the memory chips could be faulty.

Having already spent considerable time on the board and not having the materials for proper diagnostic tests, it was set aside. Work continued with a board from the MSI 6800 microcomputer in the lab built by Brian Wong.

4.4 MC6850 ACIA device

To write programs to handle data input/output (I/O) via a serial port, a knowledge of the working of the MC6850 Asynchronous Communications Interface Adapter (ACIA) is required. The ACIA allows communication between the MPU (Microprocessor Unit) and peripheral devices like printers and monitors. The ACIA permits data to be transmitted in a serial format along one line, one bit at a time under control of a "baud rate clock". For example 300 baud would mean transmitting data at 300 bits/sec (30 char/s for 1 stop bit). The ACIA receives data from the MPU in the parallel form, i.e; eight bits together, but it transmits in a serial form.

The ACIA has four 8 bit registers (i) Transmit Data Register, (ii) Receive Data Register, (iii) Status Register and a (iv) Control Register. The Transmit Data Register holds the data from the MPU until it is transferred. Receive Data Register holds the data that is transferred from a modem or a peripheral device to the ACIA. The Status Register maintains the current condition of internal ACIA activities. This register can only be read by the MPU. Each bit of the Status Register provides unique information. For example, while the MPU is waiting for data it checks the contents of bit 0 to see if new data has been received. If the bit is set, i.e at 1, it indicates that the Receive Data register is full. If the bit is not set, i.e it is 0, it indicates that no new data has been received since initialization, or the last character has been read in.

If the MPU is transmitting data to an external device it checks to see if the contents of the Transmit Data Register have already been transferred, and is ready for more. If bit 1 is set it means the contents of the Transmit Data Register have been transferred and the next character may be written to the ACIA. If it is 0 then the contents have not yet been transferred. The Control Register is used by the MPU to control the transmitting and receiving of serial data. This is a write-only register.

The Status and Control registers share the same address location. The registers are differentiated by the Read/Write line on the CPU. The Transmit and Receive data registers also share the same location.

In the SWTPC the serial interface is a MP-2S board with a MC6850 ACIA. This board is required for transmitting and receiving data between the microprocessor and an external device. It is plugged into port 7 of the SWTPC. E070 is the hex. address of the Status and Control Register, and E071 is the address for the Data Register.

The MP-S2 board has jumpers for the transmit and receive clock rate. The jumpers were configured to 300 baud, i.e transmits and receives 300 bytes per second, for compatibility with a serial printer (Decwriter). The interrupt request line on the interface is connected to the computer system's interrupt request queue (IRQ) interrupt line. The board has two DB-25 connectors. All input lines are RS-232C compatible.

The original goal was to use the SWTPC MC6809 to get data from the Varian spectrometer and store it for future use on the disk. Some software was written in 6809 assembly. A communication program was written. This program does two things, it checks to see if there is an entry from the keyboard which it then transmits to an external device and it also checks to see if there is any new data in the receive data register in the ACIA which it stores in a file. Two other programs, to print a file and to save a file on a floppy disk, have been written. The listing of these programs are in Appendix I.

It was not possible to use the SWTPC and the Tracor

Northern to collect data when the Tracor Northern, a digitizer, malfunctioned. The cost of repair was more than that required to get an IBM compatible microcomputer and an IBM data acquisition card. Obtaining ancillary hardware as well as software would also be easier for this newer and much more common system.

4.5 The IBM compatible personal computer

It uses the INTEL 8088, a 16-bit microprocessor, in a 40-pin configuration. The microprocessor has four 16-bit general purpose registers AX, BX, CX, and DX. These can be used as eight 8-bit registers. They are used to hold intermediate results. It also has a stack pointer (SP), a base pointer (BP) and two index registers SI and DI. These are for addressing specific areas in memory. The instruction pointer (IP) is equivalent to the program counter in the 8-bit MC6809 processor. The flag register is 16-bit with both status and control registers the functions of which were discussed in the previous section.

The memory is divided into four segments of up to 64K bytes each, with each segment falling on a 16-byte boundary. The segment registers (CS-code segment register, DS-data segment register, SS-stack segment register, ES-extra data segment register) are used for memory addressing.

The analog to digital conversion of the EPR signal is accomplished by the IBM DAC (data acquisition card). It uses the AD 7545 BQ and AD 574 AKD chips [5]. For easy connection a distribution panel is provided.

The signals required for data transmission from the spectrometer to the IBM DAC card are address advance, digital common, start scan, EPR high and EPR low. The address advance output is tested in the software to increment the address every time a new data point is got. The digital common is the ground. Start scan communicates to the software to start data acquisition. EPR high and EPR low are the positive and negative voltage levels of the EPR analog signal. The interface adapter for the Varian spectrometer is the P007 which had not been installed in the spectrometer system. The signals were taken from the Graphtec plotter (Y-axis) and the Bruker field controller (X-axis). The Varian spectrometer was, thus, treated as a Brucker spectrometer when using the EPRware (more details given later).

The analog EPR signal from the spectrometer was delevered to the Y input terminals of the Graphtec plotter. Therefore the required "EPR high" and "EPR low" were taken from the Y(+) and Y(-) on the Graphtec plotter. The address advance and start scan were taken from the field-controller unit. Both were connected to the address advance of the field controller unit and the digital common was chosen to be the plotter ground (see fig. 1.2 in chapter I).

Data acquisition starts when a positive going pulse is received by the DAC card. The address advance outlet is a

BNC female connector and delivers a positive going pulse of width 15 μsec when changing from sweep address to sweep address ± 1. Shielded cables had to be used, without which the signal would be hardly discernible from the surrounding noise.

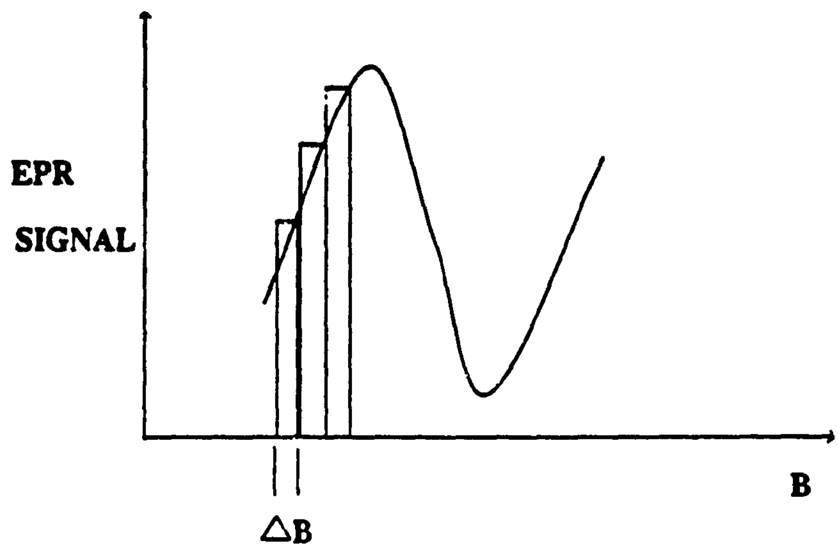
4.6 EPRWARE - software for data acquisition

Once the data is generated and collected in files and stored on a hard disk or diskette, it can be accessed whenever needed. Here the data being collected is from an EPR spectrometer. The software that accomplishes it is the EPRware developed by Dr. Philip D. Morse. It is menu-driven and allows data acquisition from both Varian and Bruker EPR spectrometers. The software runs on an IBM personal computer and requires an analog/digital card built by IBM. The software is capable of collecting data during a sweep of the magnetic field. Data are collected at a speed of 50 μsec/point and stored in 1024 point arrays. EPRware allows the user to have many arrays for the storage of data. The limitation on the number depends on the size of the array.

Data acquisition starts when a positive going pulse is received by the software. This pulse is derived from the Bruker field controller as explained earlier. The software calculates the fastest time required for the utilized computer to collect 1000 points. It does this by a routine that checks the speed of the computer. The IBM-PC compatible of frequency 8 MHz with a Intel 8088 microprocessor takes 5.115 minutes for 1024 points. The array size can be changed, and the corresponding minimum scan time required is displayed on the monitor. Data acquisition is terminated when the array is full. The data is then displayed on the monitor. If the center field and field sweep are entered in the comment section of the array that holds the data, the field values can be calculated corresponding to each data point. The array size one chooses depends on the steps per scan made by the recorder. If the size is smaller some data is lost. The size of the arrays needed for the Brucker and Varian spectrometers vary. 50μs are required to collect one point; therefore quite a few points are averaged to get a single datum. Data points are collected during one pulse cycle of the address advance line (mentioned in the previous section). On the low to high transition of the pulse these points are averaged to give a single datum point that is stored in the ax register. The address advance pulse width varies with the field scan time (t) that is chosen. To explain the above more clearly consider the array size to be 1024, field sweep B to be 3000 gauss and the scan time t to be 120sec. If B is the field sweep then the change in B for one address advance pulse is ΔB in a time Δt. Since at the end of one pulse one datum point is acquired for 1024 data points 1024 pulses should elapse. Then change in field is given by;

ΔB = B/array size = 3000/1024 = 2.93 gauss and Δt = t/array size = 120/1024 = 0.117 sec

0.117 sec is the time for acquiring one datum point. The net effect is that the EPR spectrum is displayed as a histogram as shown in fig. 4.6.



-FIG. 4.6 HISTOGRAM PRODUCED BY ANALOG TO DIGITAL CONVERSION OF THE SIGNAL.

The g-values are calculated if the comment array contains the field center and microwave frequency. The g-value is (714.448 x microwave frequency)/magnetic field at that point.

The software allows one to choose to display the data on the spectrometer plotter, spectrometer oscilloscope, computer monitor; or a digital plotter. Files are stored on disk as ASCII files and can be called by Fortran, Basic, Forth or Pascal programs. The operation of EPRWARE is described in Appendix III.

The software also allows data manipulation. Listed below are some of the utilities:

(i) Addition of a constant to array data, which allows the spectrum to be displayed higher or lower on the screen.

(ii) Multiplying the data by a constant to scale it.

(iii) Subtracting one array from another.

(iv) Baseline adjusting - required before integration.

(v) Integration: this is done by first finding the range of the data and then scaling this to the largest integer the computer can have (32767). Each Y value (EPR data) is added on to the previous to obtain the sum at the given field value. The interval between which data has to be integrated can be changed using the left and right cross hairs (seen while running the program).

(vi) Horizontal offset, which is used to align spectra.

(vii) Expanding a data file. This is done if one wants to align spectra which have been taken at different sweep widths.

(viii) Smoothing a data file.

(ix) A data point can be changed by using the key board (Glitch correction) to move the cross hair to the position of the point that has to be changed.

(x) Double integration.

The current version of the program is in the 8086 machine code, having been compiled in Quick Basic a Microsoft software product [6].

CHAPTER V

ANALYSIS OF DATA

The following chapter gives details of a few experiments carried out to test the computerized system of data recording.

5.1 Calculation of g_x, g_y, g_z, for $YBa_{1.9}Na_{0.1}Cu_3O_{7-\delta}$

The above sample (hereafter referred to as YBaNaCuO) is a high T_c superconductor with the onset of superconducting transition at 91K [10]. The structure of $YBa_2Cu_3O_7$ is perovskite like [7] therefore that of YBaNaCuO (see fig. 5.1) should be the same, since the only difference between the two is the replacement of a percentage of Ba by Na. In the superconducting phase the symmetry is orthorhombic. Fine measurements of electrical resistivity show two critical temperatures 90.5K and 79K [8]. The sample was prepared by first mixing stoichiometric amounts of CuO, Y_2O_3, $BaCO_3$, with adequate proportion of $NaHCO_3$, and grinding in a ball mill. The mixture was then heated and calcinated at 800^oC for 12 hrs. The resulting product was reground, pressed into a pellet and sintered at 960^oC in oxygen gas for several hours. This was then cooled to room temperature [8].

The sample was in the form of a small black pellet. This was stuck to the glass rod and placed in the cavity. The EPR spectrum of Cu^{2+} was observed at room temperature.

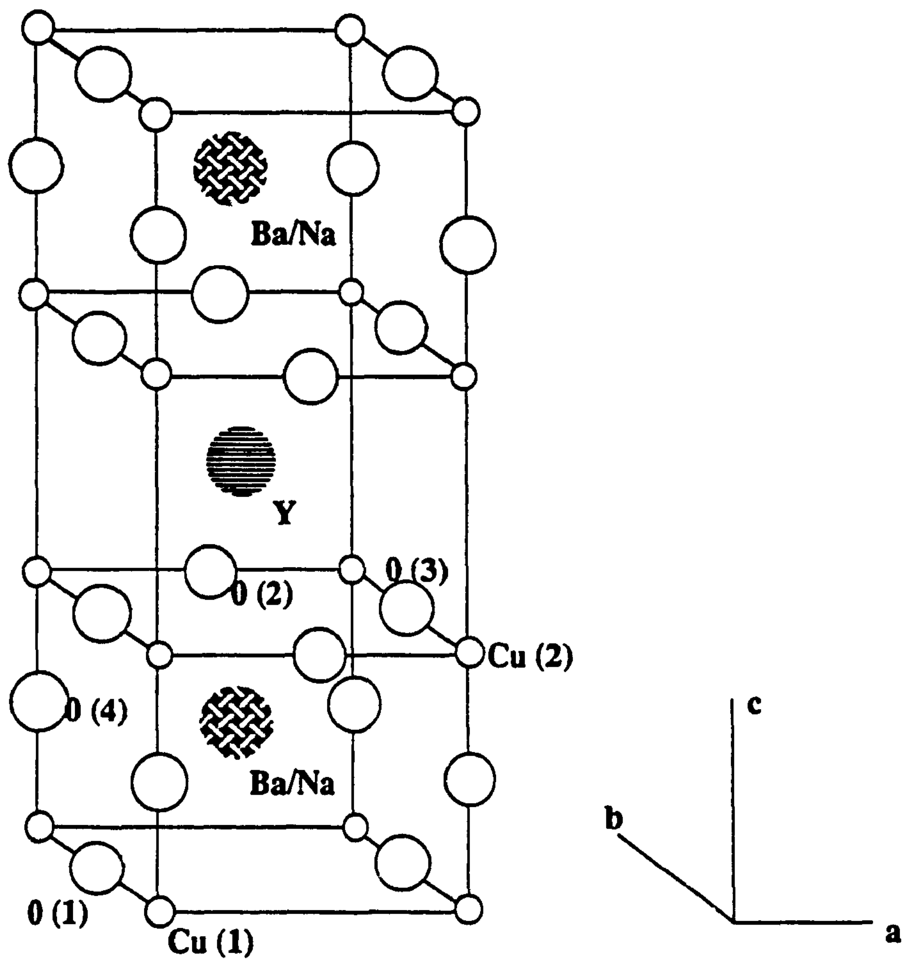


FIG. 5.1 STRUCTURE OF YBaNaCuO

The EPR Cu^{2+} signal is due to the presence of the phase with increased oxygen vacancies, characterized by an increased number of localised electrons of the Cu^{2+} ions. It is known from literature that the superconducting phase does not exhibit any EPR signal. The local symmetry about the Cu^{2+} ion in the phase of the sample with increased oxygen vacancies, responsible for the Cu^{2+} EPR signal, is deduced to be orthorhombic, as revealed by the Cu^{2+} principal values (g_x, g_y and g_z). It is concluded that the monovalent Na^+ ions substitute for Ba^{2+} ion, and not for Cu^{2+} ions. This is because if Na^+ ions replaced Cu^{2+} ions T_c would have been lowered and this has not been observed for the sample.

The EPR line of YBaNaCuO has multiple peaks. A plot of the digitized data is shown in fig. 5.2. The g-values are calculated using the line positions and are denoted in fig. 5.2 as g_z, g_x, and g_y. To separate the component peaks the EMSN 7470, a software package that can find any number of component peaks of a Gaussian or Lorentian if specified, was used (given by Dr. Capobianco, Chemistry Department, Concordia University). It gives the line position and the line width of each curve. To use the program the data file had to be in a specified format. The "mkfile.exe" (listing in Appendix II) program collects the data from the EPR data file between the stated points and stores it in a new file according to format. The first line contains the initial field value, the final field value and the step size, then

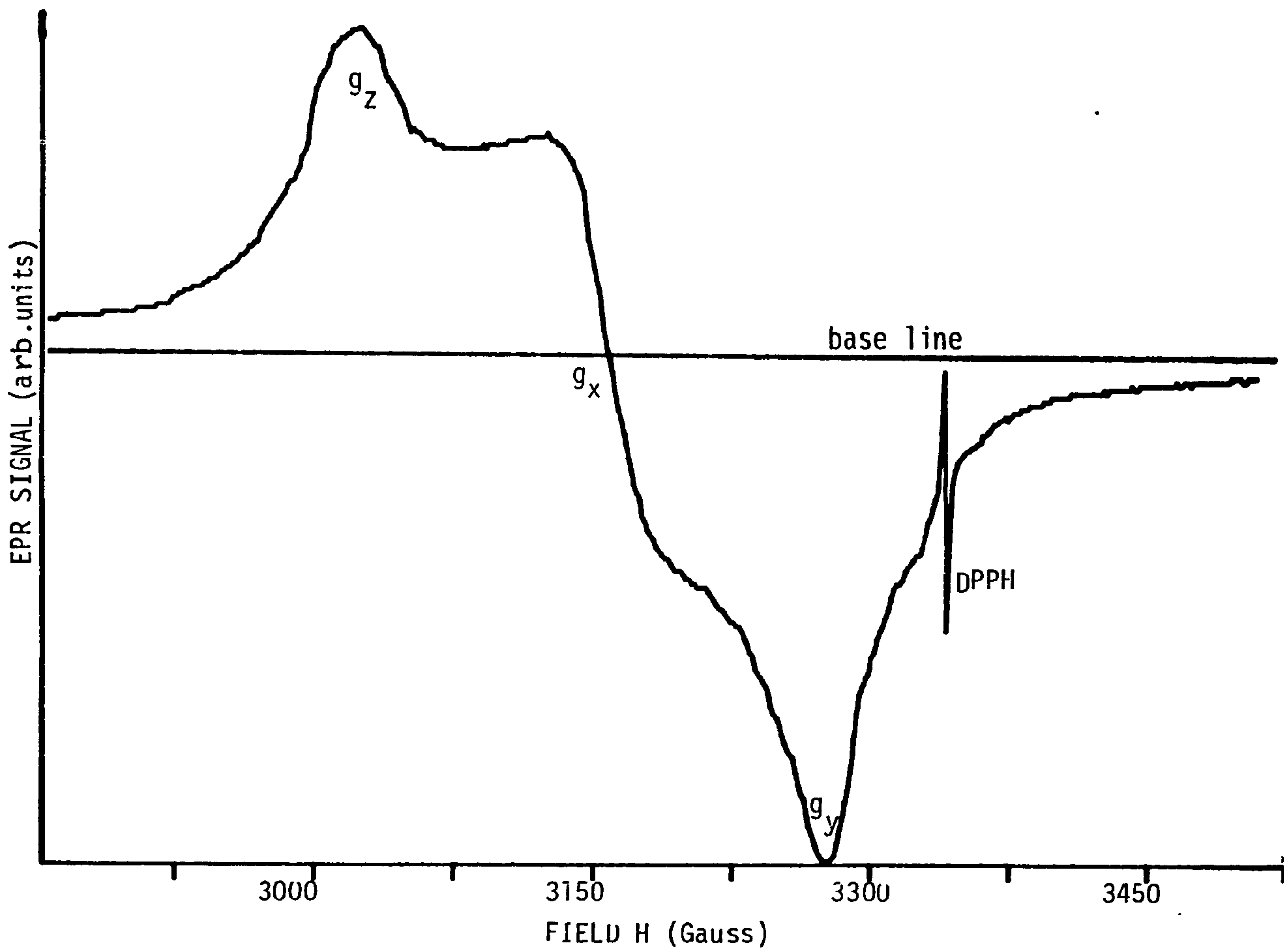


FIG. 5.2 PLOT OF THE YBaNaCuO SPECTRUM FROM DIGITIZED DATA AT ROOM TEMPERATURE

all the data points are listed one after the other. The last line has the minimum and maximum values of the data points. The g factors were calculated from the resonant field positions. The EMSN 7470 software package requires the data points to be positive, i.e above the baseline. So the signal was split into two halves, above and below the baseline which is shown in fig. 5.2. The fitting of the curve above the baseline was straight forward since the data points were positive, see fig. 5.3a. The portion below the baseline comprised of negative values. This was made positive by adding a constant vertical offset to all the data points. This was then fitted using EMSN 7470 and the resulting curve and the component peaks are shown in fig. 5.3b. The curve marked as "A" are the data points, "B" is the fitted curve "C" and "D" are the component peaks. The field positions are estimated by the software and shown below each peak along with the width.

H_z= 3023.87 Gauss;

H_x= 3175.86 Gauss;

H_y= 3289.36 Gauss;

ν = 9.5345 GHz;

Using the formula;

$$g_\alpha = h\nu / \mu_B H_\alpha \quad (\text{where } \alpha = X, Y, Z) \tag{V.1}$$

where h is the Planck's constant and μ_B the Bohr magneton. The g_z, g_x, g_y values were calculated, using eq.(V.1), to be 2.25, 2.125, and 2.07 respectively. The determined values of g_z, g_x, g_y in YBaNaCuO sample are 2.2275, 2.1180,

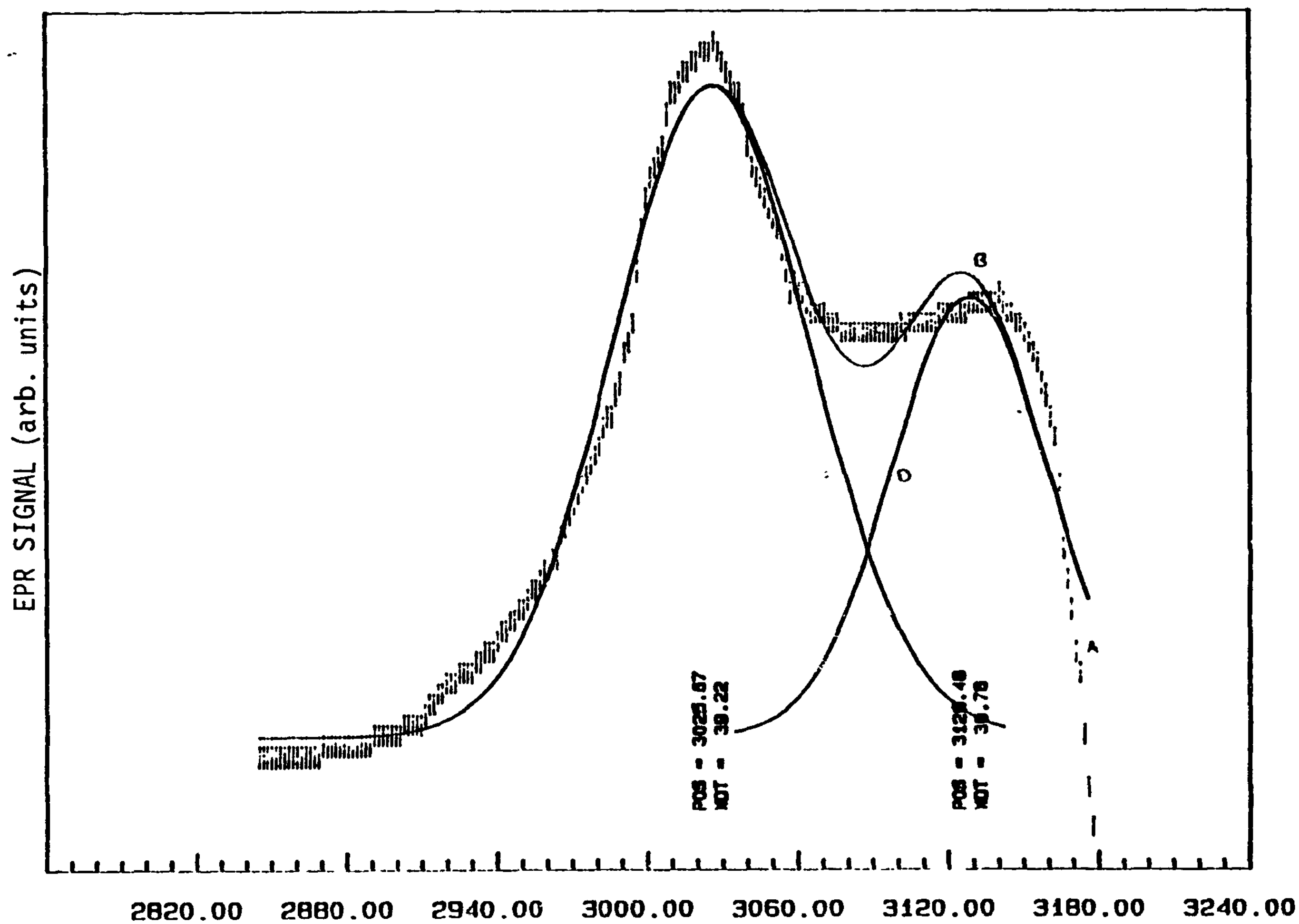


FIG. 5.3a FITTED CURVE OF YBaNaCuO (above the baseline)

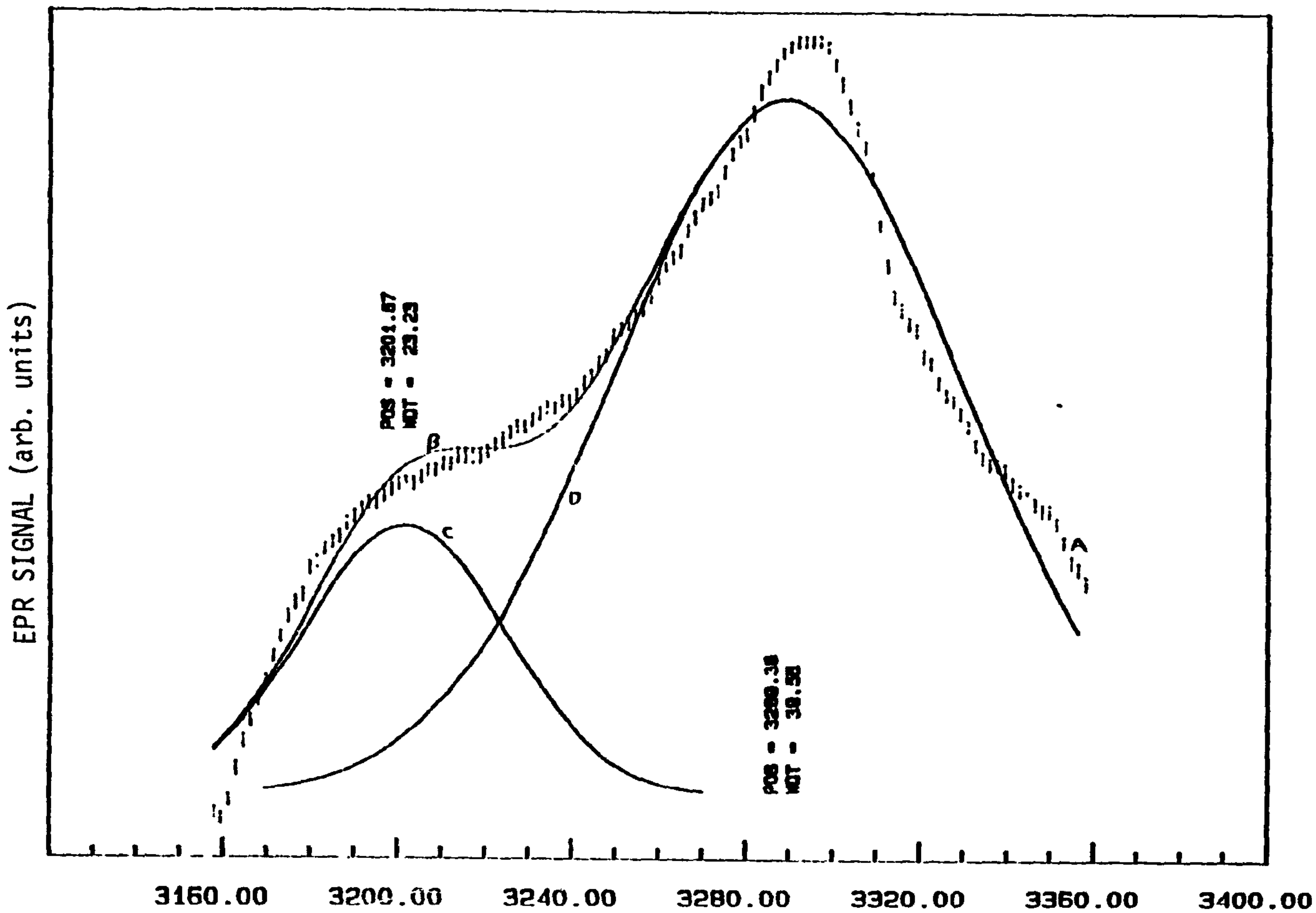


FIG. 5.3b FITTED CURVE OF YBaNaCuO (below the baseline)

2.0460 from Q-band spectrum [10] exhibiting, orthorhombic local symmetry around the Cu^{2+} ions; this is in accordance with the observation of Kobayashi et al [9]. They got g_z ,g_x and g_y for $YBa_2Cu_3O_{7-\delta}$ at room temperature to be 2.22, 2.11 and 2.04 respectively. The EPR line shape is due to the Cu^{2+} ions. Therefore one can compare the present g-values for the sample YBaNaCuO with comparable values reported in the literature for a certain phase of the $YBa_2Cu_3O_{7-\delta}$ sample.

5.2 Intensity variation with temperature for $CdCl_2.2NiCl_2.12H_2O$

From preliminary investigation and crystal observation the sample was found to be a hydrated double salt, piezoelectric, green in colour and having P_3 symmetry.

The crystal was placed in the cavity and the temperature was lowered from 300K to about 123K. The temperatures below room temperature were maintained at the sample by passing nitrogen gas through coils immersed in liquid nitrogen. Spectra for the sample at different temperatures were obtained and stored in files on the computer. Figure 5.4a shows plots of digitized data of the sample at different temperatures. The integrated intensity was calculated by using the integration utility in EPRWARE. The integrated intensity decreased with increasing temperature (see fig.5.4b). The increase in intensity is due to an increased Boltzman population difference between

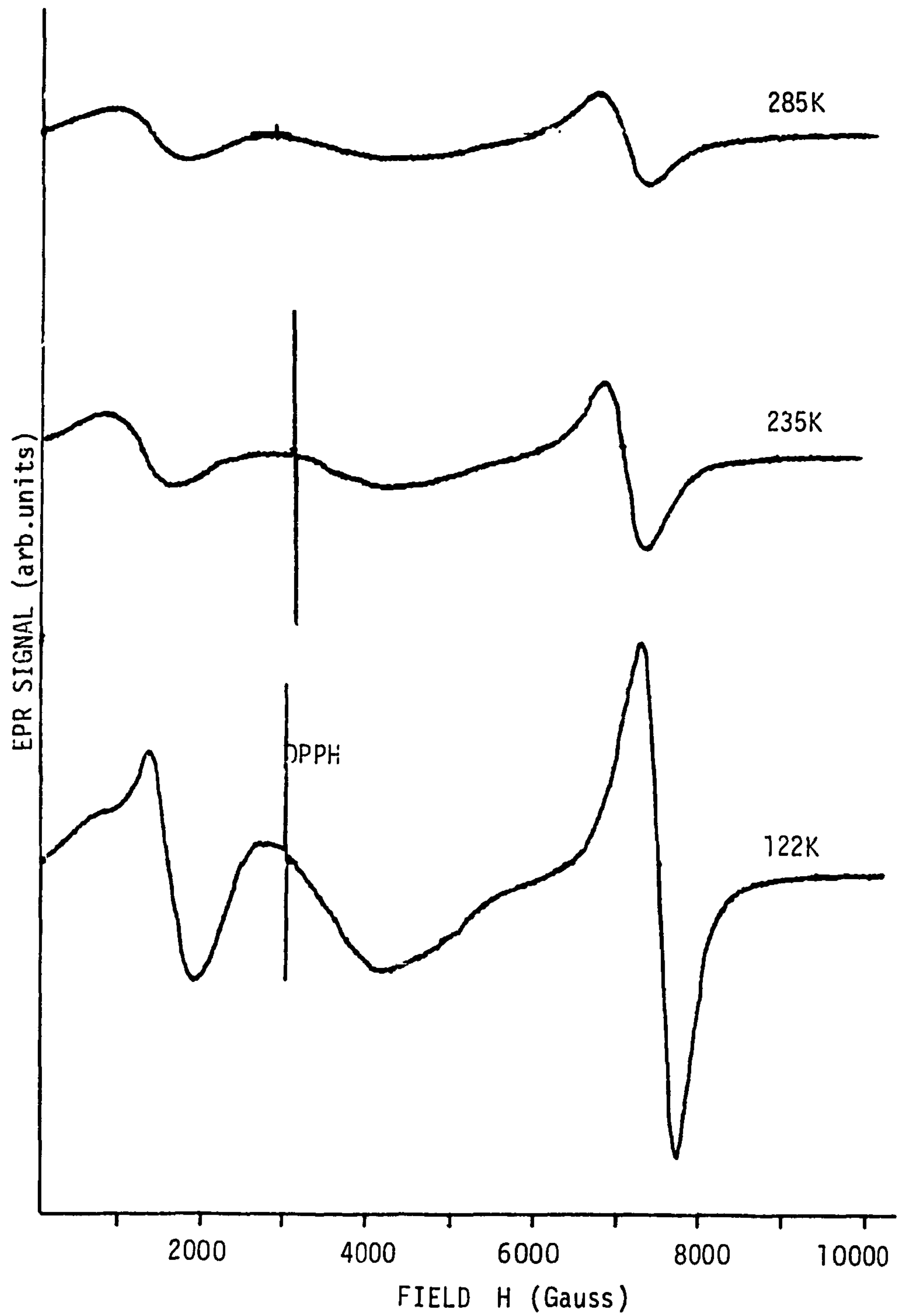


FIG. 5.4a SPECTRA OF $CdCl_2.2NiCl_2.12H_2O$ FROM DIGITIZED DATA

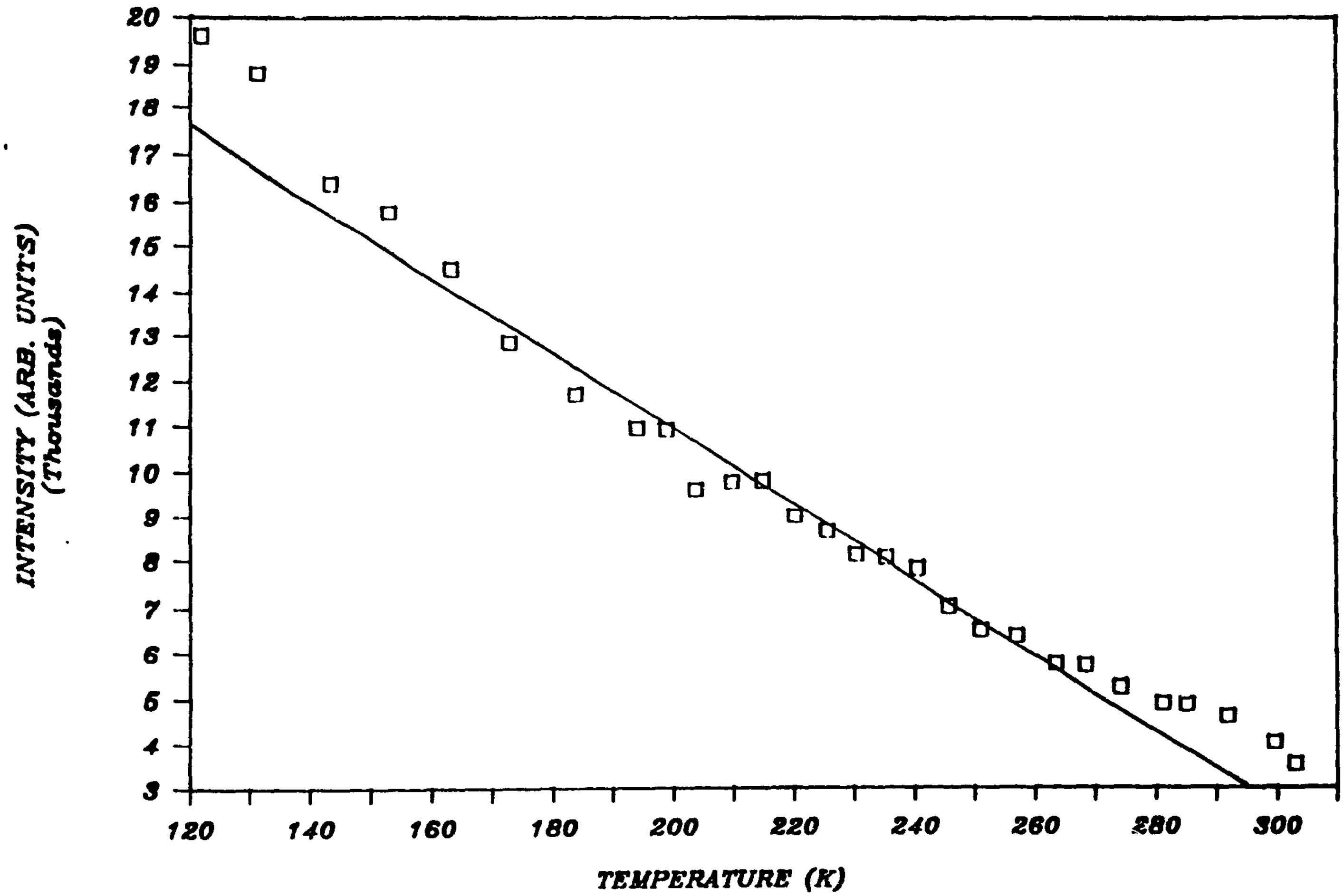


FIG. 5.4b VARIATION OF INTENSITY WITH TEMPERATURE FOR $CdCl_2.2NiCl_2.12H_2O$

the two energy levels participating in resonance [14].

5.3 Intensity variation with temperature for $Al_{55}Si_{25}Mn_{20}$

Rapidly quenched Al-Si-Mn have been reported to have icosahedral symmetry. These substances are intermediate to crystalline and amorphous materials. The concentration of Mn in Al-Si-Mn determines the formation of an icosahedral phase. If the number of atoms of Mn in the formula is more than 24 it does not form an icosahedral phase [15]. Larger concentration of Si lead to formation of an amorphous phase. X-ray diffraction of icosahedral Al-Si-Mn show that it is well ordered structure. Dunlap et al [16] have reported the existance of a ferromagnetic phase below about 120K.

The sample of $Al_{55}Si_{25}Mn_{20}$ was made by induction melting of Al, Mn and Si of very high purity. This is done in an atmosphere of argon in a boron-nitride crucible [15]. The alloy was rapidly quenched from melt on copper rollers. X-ray diffraction measurements showed that the ribbons were amorphous. he icosahedral sample was prepared by precipitating this phase from the amorphous phase by annealing as suggested by Dunlap et al [16].

The annealed and unannealed (amorphous) samples of $Al_{55}Si_{25}Mn_{20}$ were placed in the cryostat individually and cooled to a temperature of 80K using liquid nitrogen and nitrogen gas. Next the temperature was increased to 120K. The Magnetic field was swept in the range 0-600mT and a modulation frequency of 400Hz was used. The samples were suitably prepared (dimensions of about 3mm by 2mm by 0.5mm) in order not to overload the cavity.

The effective isotropic g-values of both i-phase (icosahedral phase) and a-phase (amorphous phase) are temperature dependent. There is a large variation of the g-values as the temperature is raised from 4.2 to 123K [11]. This is attributed to the temperature dependent magnetisation in the ferromagnetic phase [12].

The integrated intensity of the EPR absorbtion line was calculated for both the amorphous and icosahedral samples. The integrated intensity of the EPR line, $\mathcal{I}$, is represented by the area under the first derivative EPR absorption line. Susceptibility (χ), given by $\chi=(T_c-T)^{-\gamma'}$ for $T<T_c$ [13, 17], is proportional to intensity. This implies that there is a relationship between $\mathcal{I}$ and T_c, $\mathcal{I}=(T_c-T)^{-\gamma'}$. The following paragraphs explains how the integrated intensity and hence $-\gamma'$ and T_c were calculated for the a-phase and i-phase samples.

All the spectra for the various temperatures were stored on computer files. A few spectra for different temperatures for both the samples are shown in figures 5.5a and 5. 5b. The line shape was extracted from the noise by smoothing the data. The smoothing had to be done a couple of times for those cases for which the noise was excessive. The area was then calculated by integration. Both the smoothing and integration were done using the utilities in

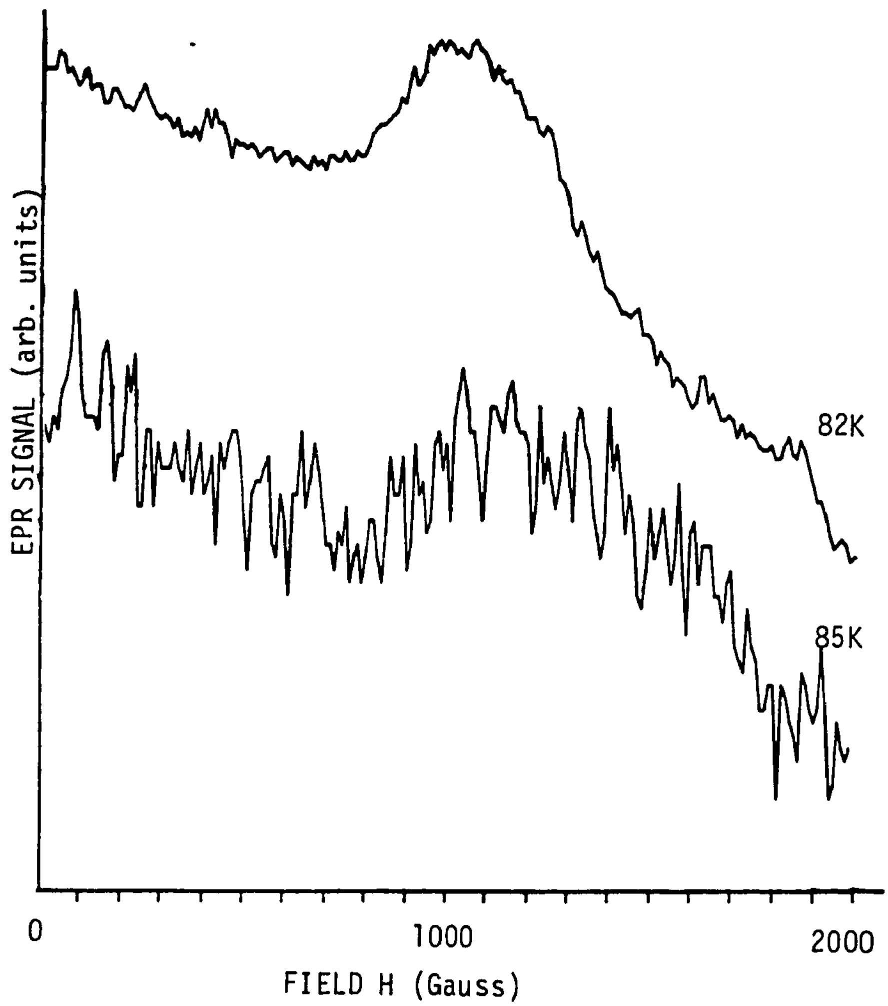


FIG. 5.5a SPECTRA OF AMORPHOUS $Al_{55}Si_{25}Mn_{20}$ FROM DIGITIZED DATA

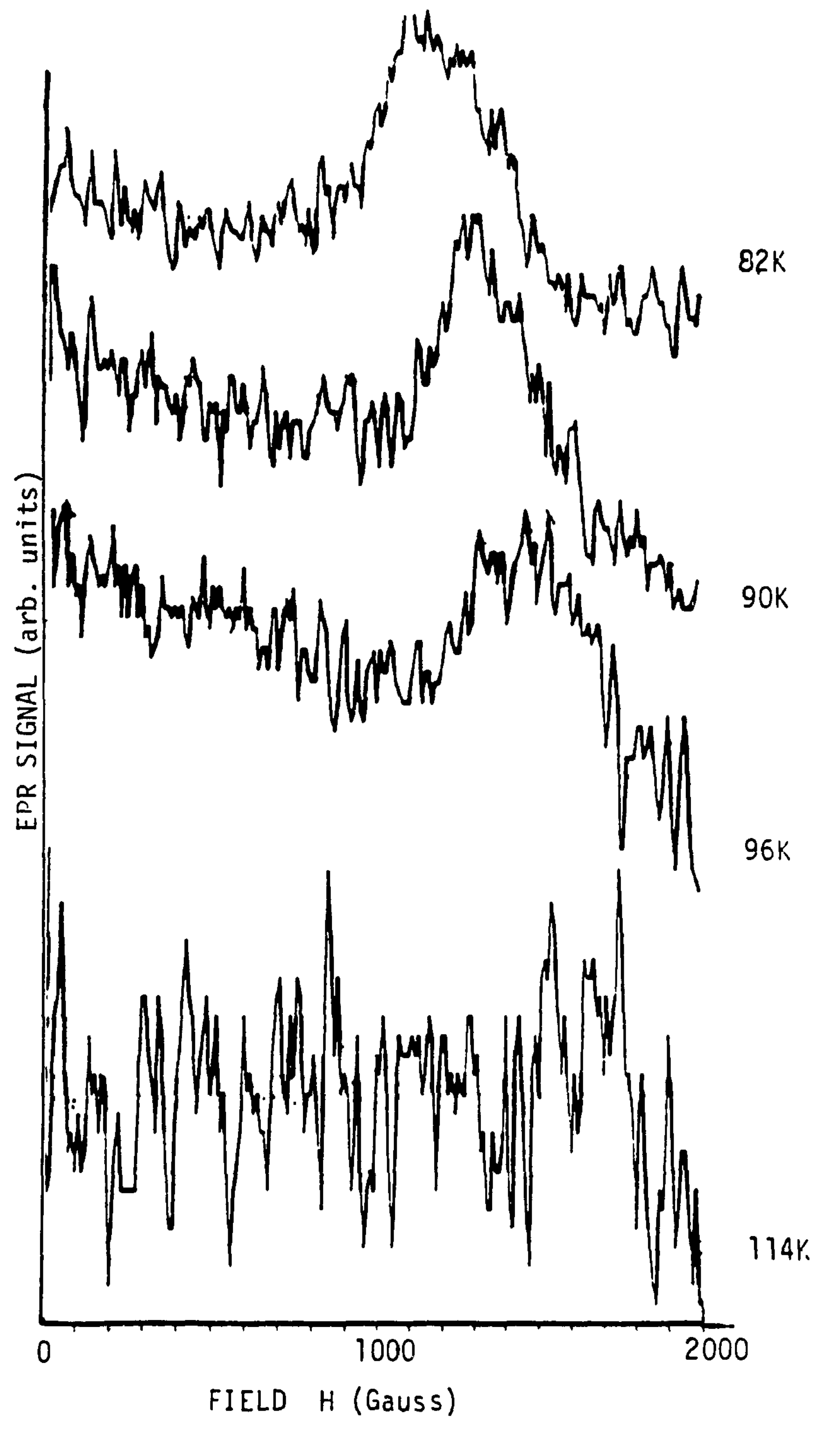


FIG. 5.5b SPECTRA OF ICOSAHEDRAL $Al_{55}Si_{25}Mn_{20}$

FROM DIGITIZED DATA

the data manupulation menu of the EPRWARE software. Before integrating the peak had to be shifted up so that the lowest point was a zero. This made it easier to calculate the area of the unwanted triangle in figure 5.6 under the line. The Y is the digital value of the EPR signal at the point H_1 (field), ΔH is the change of H from H_1 to H_2, where H_1 is the beginning of the absorbtion peak and H_2 is the end. The unwanted area was due to the fact that the base line of the spectrum was inclined at an angle to the X-axis of the screen. The plots of integrated intensity versus temperature showed that the intensity decreased with increase in temperature for both the samples (see fig. 5.7a and fig. 5.7b). By fitting the values of intensity initially to a straight line the intercept of the line with the X-axis gave an approximation of the critical temperature. At critical temperature intensity is zero. It was about 114K for the amorphous sample and 116K for the icosahedral sample. Using this as the initial value of the critical temperature (T_C) a graph of log $\mathcal{I}$ verses log of (T_C- T) was plotted for both samples where $\mathcal{I}$ is the intensity in arbitrary units (see fig. 5.8a and fig. 5.8b). The data points were fitted to a straight line. The slopes of the graphs provide initial values for γ'. γ' was about 0.72 for the amorphous sample and 0.48 for the icosahedral sample. By using the initial values of γ' and T_C a least square fitting method was used to generate the best values of γ' and T_C for both samples. For the a-phase sample

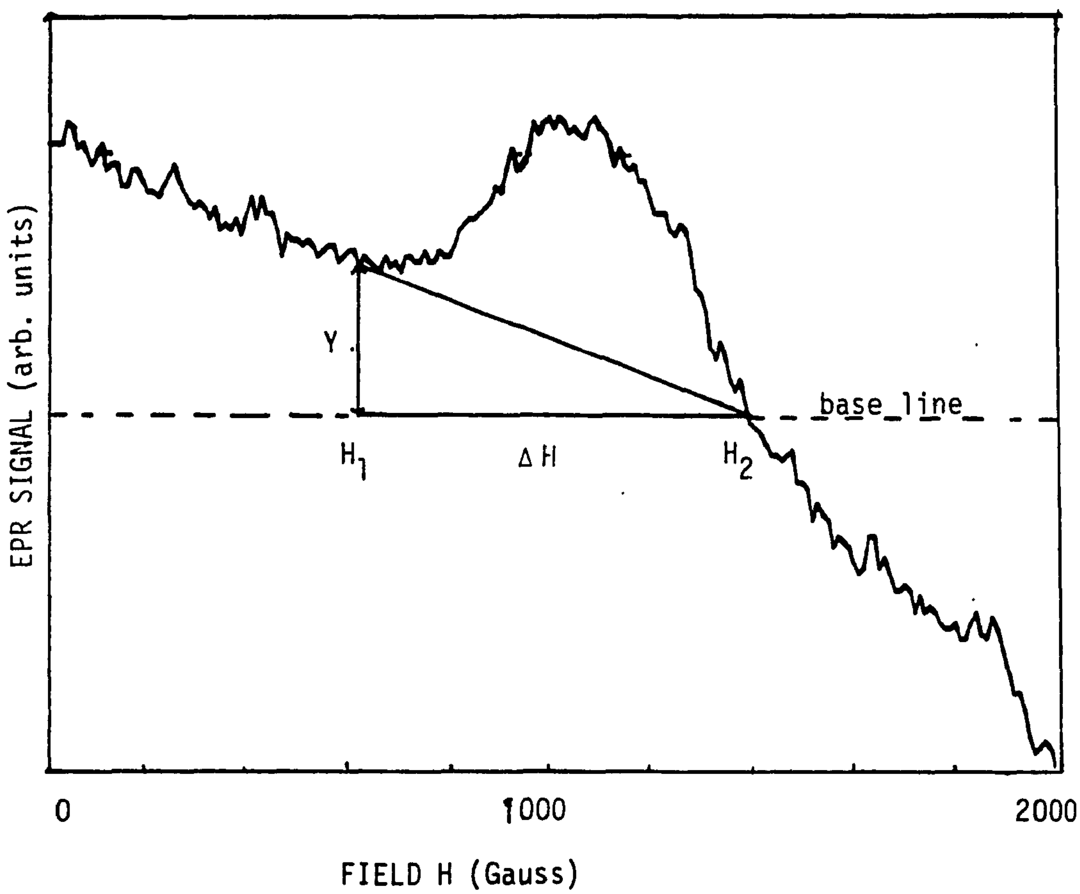


FIG. 5.6 SPECTRUM OF AMORPHOUS $Al_{55}Si_{25}Mn_{20}$, EXHIBITING THE TRIANGLE WHOSE AREA SHOULD BE SUBTRACTED FROM THE AREA UNDER THE PEAK

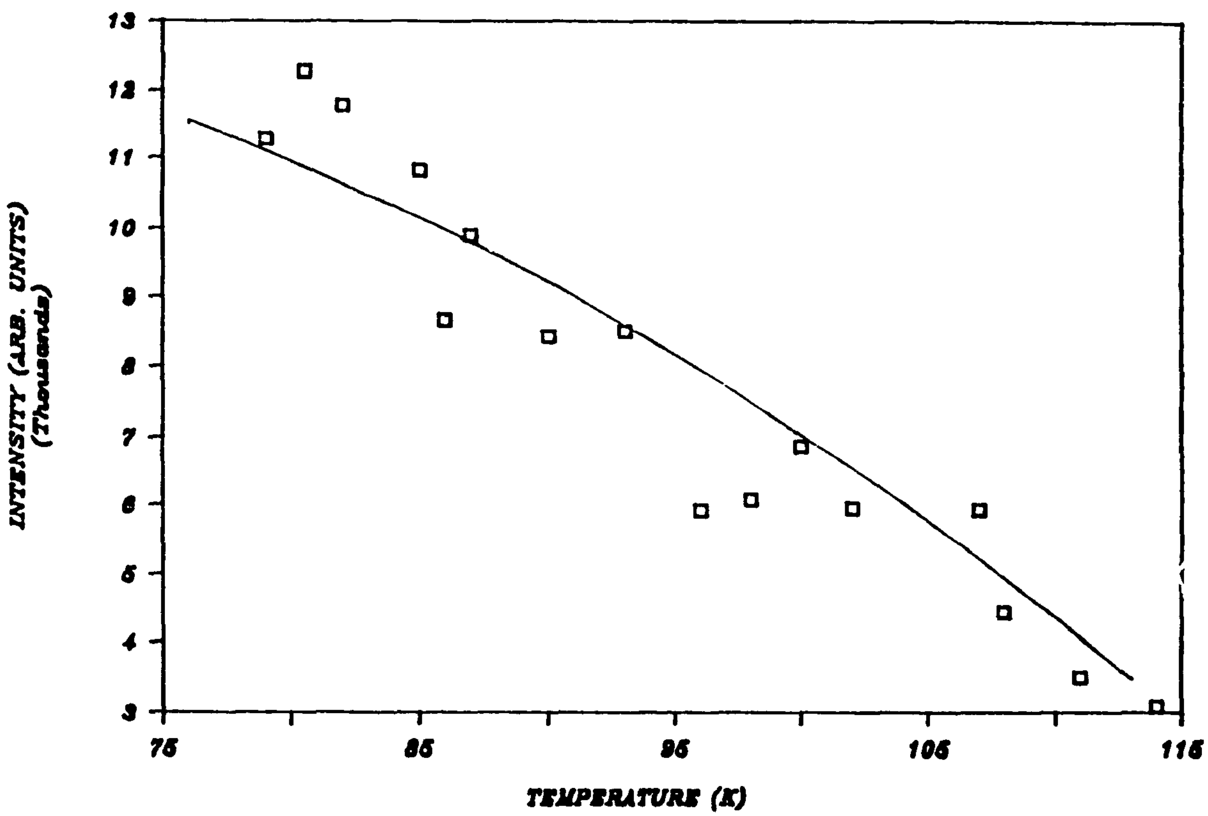


FIG. 5.7a INTENSITY VARIATION WITH TEMPERATURE FOR ICOSAHEDRAL $Al_{55}Si_{25}Mn_{20}$

The continuous curve represents the least-square fitted curve.

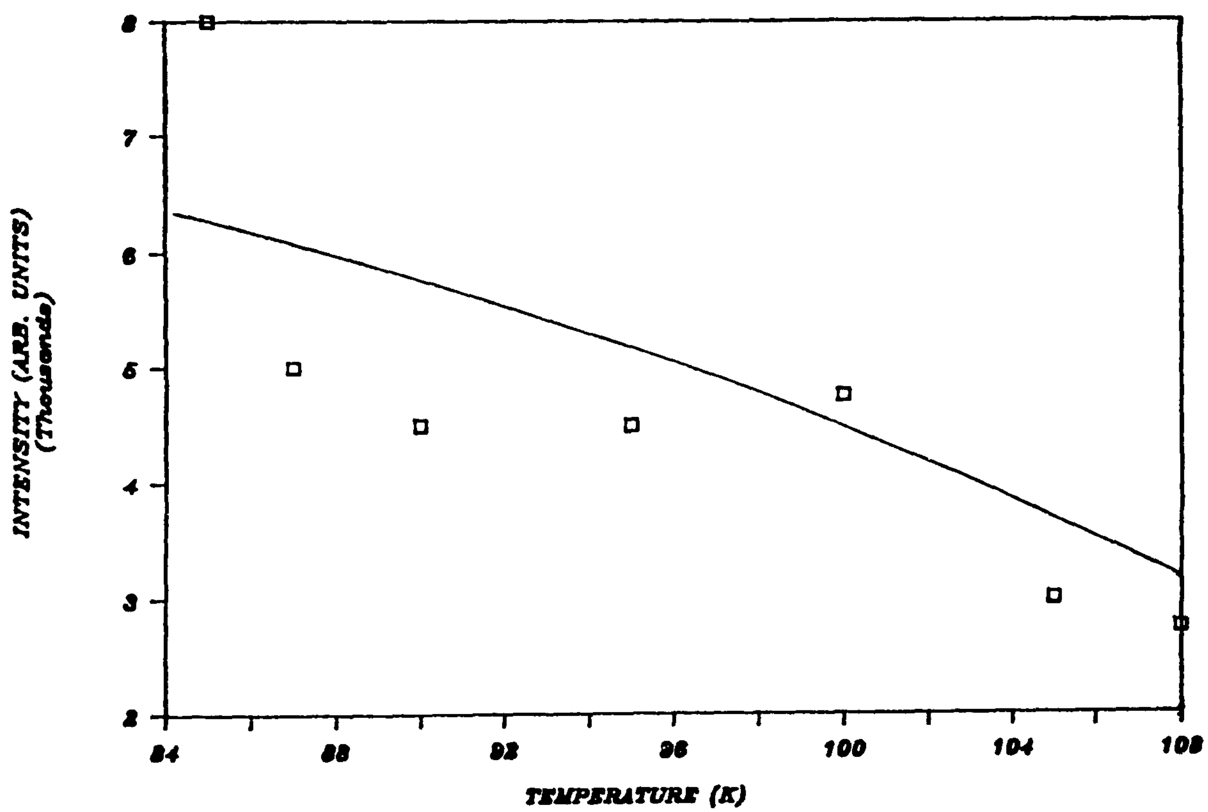


FIG. 5.7b INTENSITY VARIATION WITH TEMPERATURE FOR AMORPHOUS $Al_{55}Si_{25}Mn_{20}$

The continuous curve represents the least-square fitted curve.

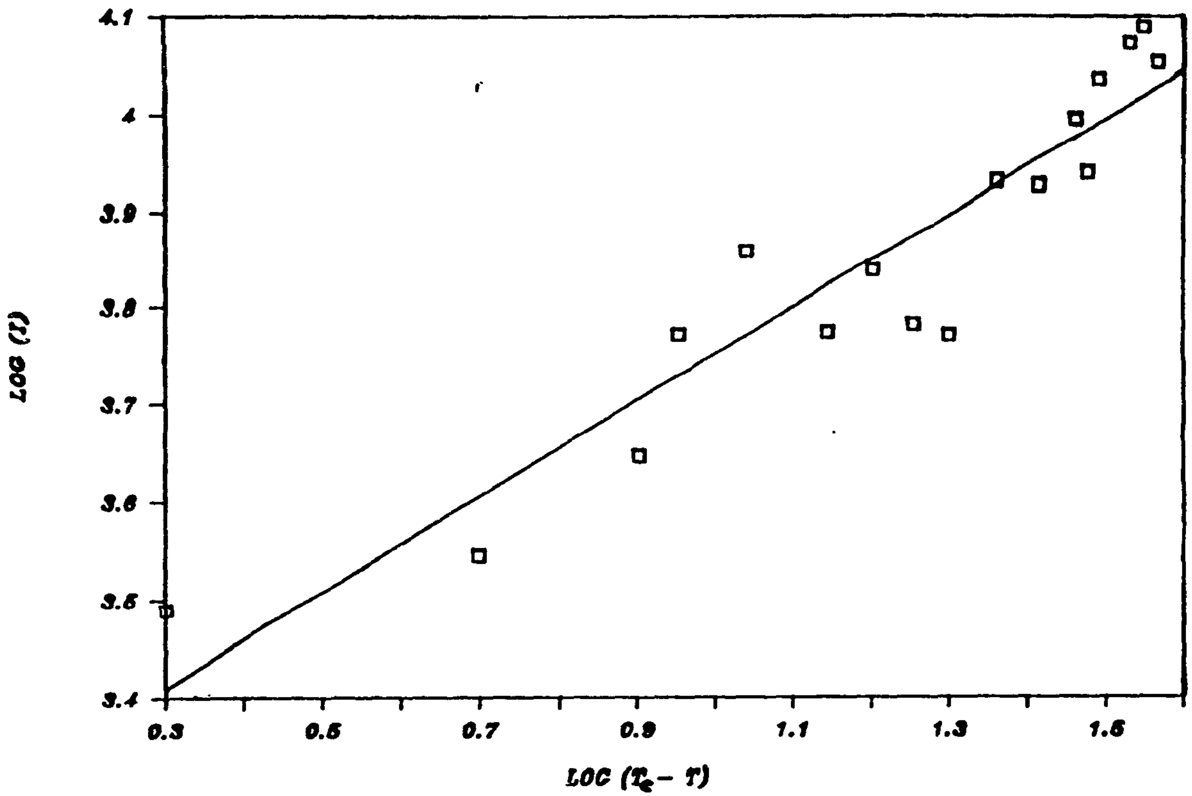


FIG. 5.8a LOG-LOG PLOT OF TEMPERATURE DEPENDENCE OF INTENSITY FOR ICOSAHEDRAL $Al_{55}Si_{25}Mn_{20}$

The continuous line represents the least-square fitted straight line.

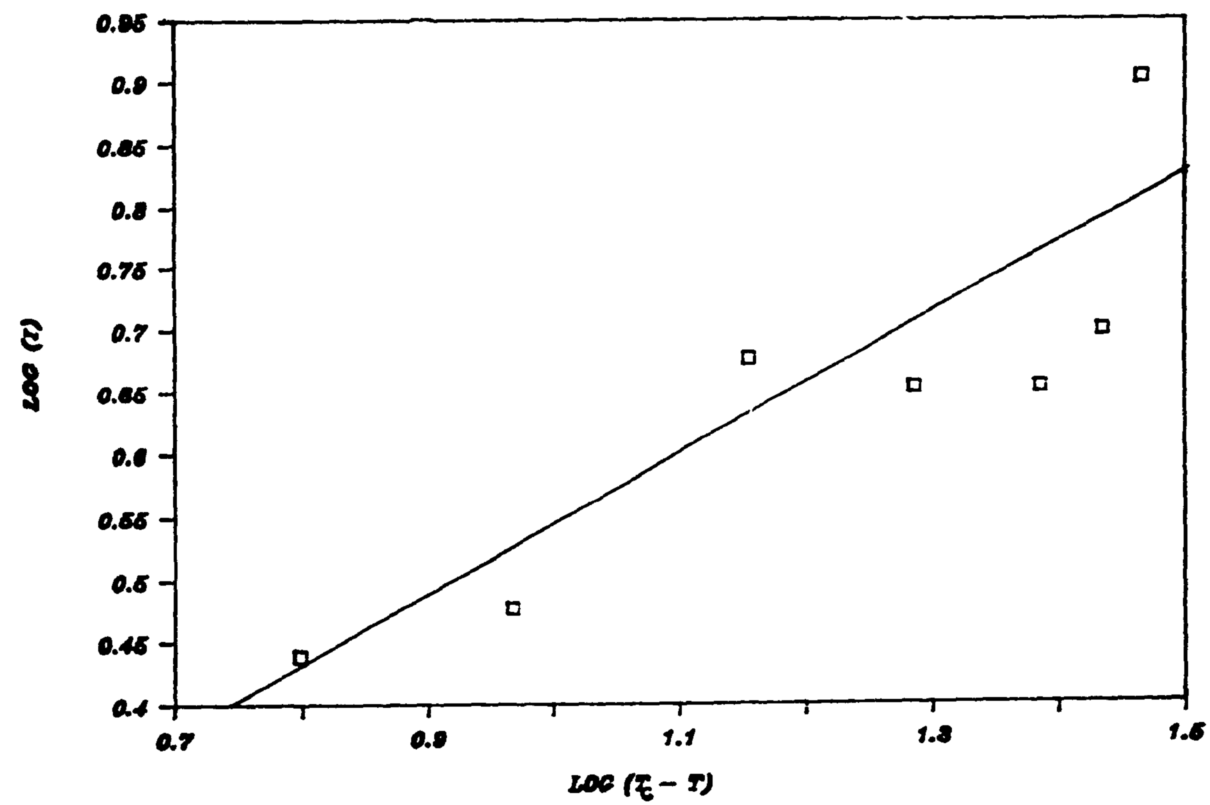


FIG. 5.8b LOG-LOG PLOT OF TEMPERATURE DEPENDENCE OF INTENSITY FOR AMORPHOUS $Al_{55}Si_{25}Mn_{20}$

The continuous line represents the least-square fitted straight line.

$\gamma' = 0.53$ and $T_c = 117K$. For the i-phase sample $\gamma' = 0.689$ and T_c=114K (see table I). Using these values intensity was again fitted to get the curves in fig. 5.7a and fig. 5.7b. The T_c values are close to what was found by Dunlap et al [16] . J. J. Hauser et al [15] found T_c of amorphous $Al_{50}Si_{30}Mn_{20}$ to be 120K. Magnetic susceptibilities for a sample of icosahedral and amorphous sample are found to be similar. The small differences are explained by the arising of intrinsic or extrinsic ferromagnetism [14]. Differences in susceptibility could be the reason for the differences in the critical parameters. Table I shows the critical temperatures and critical parameters of the a-phase and i-phase Al-Si-Mn samples.

Sample	T_c (K)	$-\gamma'$
icosahedral $Al_{55}Si_{25}Mn_{20}$	114	0.689
amorphous $Al_{55}Si_{25}Mn_{20}$	117	0.53

TABLE I. T_c and γ' for i-phase and a-phase samples

CHAPTER VI
CONCLUDING REMARKS

To automate the EPR spectrometer hardware and software were required for acquiring the data. Initially the SWTPC MC6809 microcomputer along with a Tracor Northern digitizer were used. To increase the memory of the SWTPC a 64K memory board was assembled and a floppy disk controller board was obtained to store collected data on 5.25" floppy disks. The board had to be modified slightly to work with an MC6809 microprocessor since it was configured for the MC6800 microprocessor. The programs written were in MC6809 assembly language. Software was written to obtain data from a serial port, write the data in a file on a floppy disk and to print files out on a printer. A program was also written to send data that is input from the keyboard, out to an external device and check to see if there is new data coming in through the serial port and write it into computer disk files. Due to catastrophic failure of the Tracor Northern the SWTPC system had to be put aside and an IBM compatible microcomputer with an IBM data acquisition card were used. Working with the SWTPC system gave a clear understanding of the digital electronics involved, the input and output ports, operating systems, and file management. This made it easier to understand MSDOS and the Quick Basic source code of EPRWARE.

With the data acquisition system in order data were acquired and stored in files on the hard disk for the high-T_c superconductor $YBa_{1.9}Na_{0.1}Cu_3O_{7-\delta}$, $CdCl_2.2NiCl_2.12H_2O$, and samples of amorphous and icosahedral $Al_{55}Si_{25}Mn_{20}$ alloys. For $YBa_{1.9}Na_{0.1}Cu_3O_{7-\delta}$ the principle g-values g_z, g_x, and g_y were calculated to be 2.25, 2.125, and 2.07 respectively. The intensity variation of $CdCl_2.2NiCl_2.12H_2O$ with temperature was studied. For the icosahedral sample of $Al_{55}Si_{25}Mn_{20}$ the critical temperature T_c and the critical parameter γ' were found to be 114K and 0.689 respectively, and for the amorphous sample they were found to be 117K and 0.53 respectively.

The data in digitized form can be manipulated to the desired objective, as demonstrated by the calculation of the above parameters. There is a lot of room for improving the present methods of analysis of the EPR spectrum. Writing the required software to calculate EPR line positions used in the estimation of spin Hamiltonian parameters is just one suggestion.

A problem with the automated EPR system is that the horizontal time sweep of the Bruker field controller is not synchronized with that of the IBM-PC. For example, if the sweep time has been set to 120 sec on the field controller, it does complete the sweep of the magnetic field in 120 sec but the computer does not display the data, which it should do as soon as the sweep is over. A reason for this could be that the address-advance pulse from the field controller is at the high level for 15 μsec, which is always the same, while the duration for which it is at the low level depends on the time sweep of the field controller. Thus, the pulse may be at the high level for a relatively very short period of time compared to that for the low level. The field controller sends out this address advance pulse to the computer. For each address-advance pulse coming in, the software (EPRWARE) running on the computer looks for a low to high transition to start data acquisition. The data collected during one pulse cycle, i.e, from one low to high transition to the next low to high transition, is averaged and is taken to be one data point. The software requires a certain number of data points to complete data collection. This number is specified in the software as the array size which is chosen by the user. In the present case, the array size is 1024; the software completes collection of data when 1024 data points have been obtained. Since the software gets one data point at the end of each pulse cycle, it requires 1024 pulses to get 1024 data points. After it gets all the points it will terminate data collection and display a plot of the data on the computer screen. If the sweep time chosen is five minutes in our case, all the data are displayed at the end of the sweep. This was the only period for which the software finished collecting data points in coherence with the completion of the sweep by the field controller; i.e, in this case 1024 address-advance pulses are given out by the field controller in one sweep. For sweep times greater or smaller than five minutes the computer sometimes displays data even before the end of the sweep, while at other times it does not display any data at the end of the sweep, waiting as if it needs to acquire more data to complete the collection of 1024 data points. This may be caused, among other factors, by the computer missing some low to high transitions of the address-advance pulse from the field controller to trigger data collection. It could be that some of the low to high transitions are missed from being registered by the software due to the rather small time period for which the level of the pulse is high.

Thus a solution that can be proposed is to stretch the duration for which the address-advance pulse is high, so that it has a 50% duty cycle, i.e, the low and high of the pulse are of equal durations (i.e, a square pulse). This can be accomplished by an Intel 8253 programmable timer/counter. It is known that the number of points EPRWARE needs to terminate data acquisition corresponds to the array size chosen. If the array size chosen is 1024 then all that is required is to send 1024 pulses for the sweep time (τ) that has been set on the field controller, i.e, $\tau/1024$ is the duration of each pulse cycle. This value is again divided by the period of the clock that times the Intel 8253. The result should then be fed into the Intel 8253, so that the output generates square pulses. This routine has to be executed 1024 times (the array size), and must be triggered by the start scan pulse. With the hardware and software involved in the hereby proposed solution the automation of the EPR spectrometer can be accomplished.

APPENDIX I

This appendix lists the programs written in assembly language for the MC6809 microprocessor. The programs are useful in transferring data between systems.

Dumter.txt - A communications program

The following source file "dumter.txt" is assembled with the command "ASM". It is executed with the command "DUMTER". The terminal responds to any information coming in through the ACIA and diplays it on screen. It also sends out information through a serial port on command from the keyboard.

ORG $C100	
RPTERR EQU $CD3F	* A user callable system routine that registers an error.
FCB EQU $C840	* The system's FCB area starts at this location and is a 320 byte long block of RAM.
GETFIL EQU $CD2D	* This is a user callable system routine stored at location $CD2D. It helps in getting file specifications.
FMS EQU $D406	* $D406 is the entry point for all calls to the file management system.
ACIA EQU $E070	* The address of the ACIA.
CON EQU $E004	* Address of the console port.

```
* Do master reset on ACIA
START     LDX #ACIA
          LDA #03
          STA 0, X
* Set up no parity bit, 8data bits.
          LDA #%00010101
          STA 0, X
          TST 1, X
```

* Scans the operator's console port "CON" and the modem port (external serial port) "ACIA". When input is detected from the console it is transmitted via the modem port unless it is the command ^u (CTRL u) which instigates a file upload.

```
SCAC      LDA CON
          BITA #01
          BEQ SCAM
JB        LDA ACIA
          BITA #02
          BEQ JB
          LDA CON + 1
          CMPA #$15
          BEQ FIL
          STA ACIA + 1
          BRA SCAC
* Scan modem.
SCAM      LDA ACIA
          BITA #01
```

```
* A tight loop, misses no character at high baud rates.
          BEQ SCAC
JB1       LDA CON
          BITA #02
          BEQ JB1
          LDA ACIA + 1
          ANDA #$7F
          JSR $FDDF
          BRA SCAC
FIL         LDY #$C080    * $C080-$C0FF is the line buffer.
Characters except for control characters typed on the
keyboard are placed in this buffer.
FIL1      LDA CON
          BITA #01
          BEQ FIL1
          LDA CON + 1
          STA , Y
          LEAY 1, Y
          CMPA #$0D
          BEQ PS
          BRA FIL1
* Assemble file name
PS        LDD #$C080
          STD $CC14      * $CC14-$CC15 is the line buffer
pointer. These locations contain the address of the next
character in the line buffer to be processed.
          LDX #FCB
```

```
        JSR GETFIL
        BCS ER1
* Open file to read
        LDA #1
        STA 0, X
        JSR FMS
        BNE ER1
        LDA #9
        STA 0, X
* Disk address
        LDA 17, X
LP      BSR READ
        LDX #FCB + 68
LOP     LDB #252
LOOP    LDA 0, X
        JSR PRINT
        DECB
        BNE LOOP
* Get track and sector
        LDD FCB + 64
        BNE LP
        BRA SCAC
* Print if ACIA is ready
PRINT   PSHS A
TS      LDA ACIA
        BITA #2
        BEQ TS
```

```
        PULS A
        STA ACIA + 1
* If carriage return put in line feed
        CMPA #$0D
        BEQ LFE
        LEAX 1, X
        RTS
* Send out line feed
LFE     LDA #$0A
        BRA PRINT
        RTS
* Display error
ER1     JSR RPTERR
        RTS
* Make track and sector available
READ    LDX #FCB
        STD 30, X
        JSR FMS
        BNE ER1
        RTS
END     START
```

WRITE.TXT

This program when assembled and executed as a command writes the text or data in a file in RAM whose name is specified after the write command. It then stores the file on disk. Format of the command is "WRITE <file name>".

```
ORG $C100
RPTERR EQU $CD3F
GETFIL EQU $CD2D
FMS EQU $D406
ACIA EQU $E070
FCB EQU $C840
SETEXT EQU $CD33
* Master reset on ACIA
START   LDX #ACIA
        LDA #03
        STA 0, X
        LDA #%00010101
        STA 0, X
        TST 1, X
        LDX #FCB
        JSR GETFIL
        BCS ER1
        LDX #FCB
        LDA #1
        JSR SETEXT
* Open to write
        LDX #FCB
        LDA #02
        STA 0, X
        JSR FMS
        BNE ER1
```

```
LOOP    LDA ACIA
        BITA #01
        BEQ LOOP
LP      LDA ACIA + 1
        ANDA #$7F
        CMPA #$20     * $20 is 'SP (ASCII space).
        BCC SAVE1
        CMPA #$04     * $04 is 'EOT (end of transmission).
        BEQ SAVE2
* Write one byte
SAVE1   LDX #FCB
        LDA #0
        STA 0, X
        JSR FMS
        BNE ER1
        BRA LOOP
* Write one byte
SAVE2   LDX #FCB
        LDA #0
        STA 0, X
        JSR FMS
        BNE ER1
* Close file
        LDX #FCB
        LDA #4
        STA 0, X
        JSR FMS
```

```
        BEQ EXIT
ER1     JSR RPTERR
        RTS
EXIT     JMP $CD03     * Flex warmstart (return to command
                       mode i.e. system waits for a command.
END     START
```

PS.TXT

PS.TXT when executed prints a file on a serial printer. the format of the command is "ps <file name>".

```
ORG $C100
RPTERR EQU $CD3F
FCB EQU $C840
GETFIL EQU $CD2D
FMS EQU $D406
ACIA EQU $E070
* Initialize ACIA, get ACIA address, do master reset on
ACIA
START   LDX #ACIA
        LDA #03
        STA 0, 3
* Set up no parity bit, 8 data bits, 2 stop bits
        LDA #%00010101
        STA 0, X
        TST 1, X
* Assemble file name
```

```
        LDX #FCB
        JSR GETFIL
        BCS ER1
* Open file to read
        LDA #1
        STA 0, X
        JSR FMS
        BNE ER1
        LDA #9
        STA 0, X
* Disk address
        LDD 17, X
LP      BSR READ
        LDX #FCB + 68
LOP     LDB #252
LOOP    LDA 0, X
        JSR PRINT
        DECB
        BNE LOOP
* Get track and sector
        LDD FCB + 64
        BNE LP
* Exit to flex
        JMP $CD03
* Print if ACIA is ready
PRINT   PSHS A
TS      LDA ACIA
```

```
        BITA #2
        BEQ TS
        PULS A
        STA ACIA + 1
* If carriage return put in line feed
        CMPA #$0D
        BEQ LPE
        LEAX 1, X
        RTS
* Send out line feed
LFE     LDA #$0A
        BRA PRINT
        RTS
* Display error
ER1     JSR RPTERR
        RTS
* Make track and sector available
READ    LDX #FCB
        STD 30, X
        JSR FMS
        BNE ER1
        RTS
END     START
```

APPENDIX II

The following programs are written in the C language for specific manupulation of the EPR data. They are all executable files.

To differentiate a portion of the spectra (diff.exe)

To differentiate a portion of the EPR signal received one has to type the command "diff ". The program prompts you for the file from which data has to be taken, the array size that has been used while collecting data, the initial and final array position (this can be obtained by loading the file in EPRware and going into the display menu). The F5 key toggles between displaying the array position, field values and g-values and the field sweep. The destination file name, where the differentiated data should be stored, when prompted by the software should be specified.

```
/* standard I/O library */
#include <stdio.h>
main ()
{
/* Declarations */
FILE *fp, *fopen(), *fp1;
int i, j, init, final, npts;
float step, as, epr[300], atof(), fs, f[300], eprd[300];
char *name, dummy[200];
```

```
i=0;
/* Keyboard line input */
L1: printf ("NAME OF EPR DATA FILE>");
scanf ("%s",name);
if ((fp=fopen (name, "r"))==NULL)
{
i++;
printf ("CAN'T OPEN FILE !\n");
/* Three tries to open file, then quit */
if (i<3)
goto L1;
else
exit(0);
}
printf ("ARRAY SIZE USED WHILE COLLECTING DATA =");
scanf ("%f",&as);
printf ("INITIAL DATA LOCATION =");
scanf ("%d",&init);
printf ("FINAL DATA LOCATION =");
scanf ("%d",&final);
printf ("FIELD SWEEP =");
scanf ("%f",&fs);
npts =(final - init) +1;
step= fs/as;
/* Skipping header information in file */
fgets (dummy, 200, fp);
fgets (dummy, 200, fp);
```

```
fgets (dummy, 200, fp);
fgets (dummy, 200, fp);
fgets (dummy, 200, fp);
for (i=1; i< init; i++)
fgets (dummy, 200, fp);
fgets (dummy, 200, fp);
j=0;
epr[j]= atof (dummy);
/* Stores data in epr array */
for (i=init+ 1; i<= final; i++)
{
fgets (dummy, 200, fp);
epr[++j]= atof (dummy);
}
j= 0;
for (i= init; i<= final; i++)
f[j++]= step*i;
j= 0;
/* Differentiates the data */
eprd[0]= epr[0];
for (i= j+1; i<=npts; j++)
eprd[i++]= epr[i]- epr[i-1];
/* Stores the data in the destination file as array eprd */
printf ("NAME OF DESTINATION FILE (use .DAT extension) >");
scanf ("%s",name);
if ((fp1= fopen (name,"w"))==NULL);
exit (0);
```

```
fprintf (fp1, "%f %f \n", f[j], eprd[j]);
fclose (fp1);
}
```

Program to make a data file to be used by the curve fitting program (EMSN 7470) multiple peaks-Gaussian or Lorentian (mkfile.exe).

The program is executed by using the command "mkfile". The program prompts the user to specify the EPR data file name, the destination file (where data is stored for use by the EMSN 7470 software), the field sweep, the initial and final array position and the corresponding field values.

```
#include <stdio.h>
main ()
{
FILE *fp, *fopen(), *fp1;
int i=0, j=0, init, final, npts;
float field1, field2, max, step;
float epr[1300], atof (), min, fs;
char *NAME, DUMMY [200];
/* Sends prompts to the screen */
L1: printf ("NAME OF EPR DATA FILE >");
scanf ("%s", name);
if ((fp= fopen (name, "r"))==NULL)
{
```

```
i++;
printf ("CAN'T OPEN FILE !");
if (i<3)
goto L1;
else
exit (0);
}
printf ("INITIAL VALUE =");
scanf ("%d", &init);
printf ("FINAL VALUE =");
scanf ("%d", &final);
printf ("FIELD SWEEP =");
scanf ("%f", &fs);
printf ("INITIAL FIELD VALUE =");
scanf ("%f", &field1);
printf ("FINAL FIELD VALUE =");
scanf ("%f", &field2);
printf ("NAME OF DESTINATION FILE (use .DAT extension) >");
scanf ("%s",name);
npts =final- init+ 1;
/* 1024 data values or channels */
step= fs/1024;
/* Removes part of data that is not required */
fgets (dummy, 200, fp);
fgets (dummy, 200, fp);
fgets (dummy, 200, fp);
fgets (dummy, 200, fp);
```

```
fgets (dummy, 200, fp);
/* Gets the required data into the array epr */
for (i= 1; i< init; i++)
fgets (dummy, 200, fp);
for (i= init; i<= final; i++)
{
fgets (dummy, 200, fp);
if (dummy[0]=='-')
strcpy (dummy, dummy+ 1);
epr[j++]= atof (dummy);
}
/* Obtains the minimum and maximum of the data */
min= epr[0];
j= 1;
while (j< npts)
{
if (min- epr[j++]<= 0);
else
min= epr[j-1];
}
max= epr[0];
j= 1;
while (j< npts)
{
if (max- epr[j++]>= 0);
else
max= epr[j- 1];
```

```
}
/* store data in destination file */
if ((fp1=fopen (name,"w"))==NULL);
exit (0);
fprintf (fp1,"%f %f %f \n",init,npts,step);
for (i=init; i<=final; i++)
fprintf (fp1,"%f \n",epr[i]);
fprintf (fp1,"%f %f \n",min,max);
fclose (fp1);
}
```

APPENDIX III

EPRWARE operation

EPRware acquires data from the data acquisition card and stores it in arrays that can be manipulated and stored as files.

To execute EPRware use the command EW once in the directory that contains the software.

One is then asked what spectrometer is being used. Since the positive pulse (address advance) is derived from the Bruker field controller, the spectrometer should be selected to be Bruker (B).

The software then asks what size array is required. The data array size is selected to be 1024. This is done by moving the cursor box to the required position and hitting ENTER.

EPRware is menu driven, so by moving the cursor around the menu the choices are made.

When in the main menu do the following:

a) change the drive to that which is being used to store the data collected. The default drive is B i.e. unless otherwise specified the drive to which the data is stored or read from is B.

b) the comment array for a selected array number can be filled at this point (or in the data acquisition menu) with all the spectral parameters set up for data collection.

c) from this menu one can select the next procedure to execute.

d) spectrometer type has already been set in the beginning therefore this need not be changed.

e) the file menu allows one to save, read, and delete files from the specified disc drive.

f) exit to DOS from the main menu by choosing the option and typing capital 'Y'.

g) one must make an effort to minimize disk usage, delete files as soon as use is over or archive on floppy disk and store carefully.

The data acquisition menu

This menu allows you to choose what array you want the data to be stored in. The number of scans to be made can also be specified. If more than one, the software averages all the scans before storing in the array.

The comments to be saved with the data can also be loaded in this menu. The array size can be changed. This can increase the number of points collected (results in collecting some points during the reverse scan) or decrease it (loss of some data).

The data acquired can be saved from this menu by using the " SAVE DATA TO DISK" comm . This requires the number of the array to be saved, the kind of data (g-values, field values, EPR values), and the file name to store it in.

When acquiring data, all of the following is displayed on screen:

HIT RETURN TO COLLECT DATA INTO ARRAY (ARRAY SELECTED)

DATA IN THIS ARRAY WILL BE LOST

HIT CTRL- X TO ABORT

WAITING FOR SPECTROMETER TO FINISH CURRENT SCAN OR TO BE PUT INTO REMOTE

At this stage the switch for the address advance is pressed and the spectrometer plotter is started on the scan. The following is then seen:

SPECTROMETER RETURNING TO START NEW SCAN:

When the above command is seen it is ready to acquire data when it is received.

To abort a run press CTRL-X. This works if the scan hasn't started. If the scan has started CTRL-X does not work if the field is not sweeping.

On completion of a scan the spectrum is displayed on the monitor.

From the main menu select the data manipulation menu and scale for better viewing, as described below.

The data manipulation menu

Any manipulation of data takes place between the left and right crosshairs. The crosshairs can be moved from the display menu. When in this menu one can perform operations like spectral subtraction, integration, smoothing, vertical offset, scaling, baseline adjusting, horizontal offset, expanding a data file, inputing data from the computer keyboard, moving data from one array to another, double integration and listing of selected arrays.

Display menu

This menu allows one to display the spectra on screen. This can be selected by hitting the HOME key on the keyboard. The HOME key toggles between the display menu and the previous menu.

The following is a list of the functions of the numeric keyboard and function keys in the DISPLAY menu.

UP ARROW - moves top or bottom crosshairs up.

DOWN ARROW - moves top or bottom crosshairs down.

LEFT ARROW - moves left or right crosshairs left.

RIGHT ARROW - moves left or right crosshairs right.

(Movement of the crosshair to the left and right depends on the step size. The default size is 20. Every time HOME is hit the entered step size is changed to the default value.)

PG UP - switches between left and right crosshairs.

PG ON - switches between top and bottom crosshairs.

HOME - toggles between display and any other menu.

END - selects the array whose parameters are shown in the display menu. Every time END is pressed the next consecutive array is displayed.

INSERT - adds arrays to display window.

DELETE - removes arrays from the display window.

(To get out of the INSERT and DELETE mode a negative number has to be typed and RETURN hit.)

F1 - expands the window to the size selected by the crosshairs.

F2 - returns the window to the default size vertically.

F3 - returns the window to the default size horizontally.

F4 - clears screen.

F5 - toggles between the array position, G values, and field values.

(For g-values the entry of field center, scan width and frequency in the COMMENT section are required. For field values the field center and scan width need to be entered.)

RETURN - if this alone is hit, not after any of the above functions, then the step size can be changed.

NOTE- The routine for the actual collection of data is written in assembly language for the Intel 8086 microprocessor. Definitions for the IBM DAC card are listed in the source code (EW.BAS) for EPRWARE.

REFERENCES

[1] J. Wertz and J.R. Bolton, Electron Spin Resonance Elementary Theory and Practical Applications, McGraw-Hill 1972.

[2] V-4502, EPR Spectrometer System Manual.

[3] Peripheral Tecnologies, FD-2 Controller Board Manual.

[4] Data Systems 68, Documentation.

[5] Analog devices, Data-Acquisition databook, V1, 1984.

[6] P.D. Morse, *Bio. Phys. J.* **51**, 440a (1987).

[7] D.P. Billerbach, J.R. Hardy and P.J. Edwardson, *Phys. Rev.* **B39**, 202 (1989).

[8] M. Ausloos, C.H. Laurent, H.W. Vanderschueren, A. Rulmont and P. Tarte, *Sol. Stat. Comm.* **V68**, No. 6 (1988).

[9] T. Kobayashi, H. Katsuda, Y. Sakabe, K. Iwai and S. Kanba, *Phase Transitions*, **12**, 215 (1988).

[10] S.K. Misra and L.E. Misiak, *J. Phys. Condens. Matter*, to be published.

[11] S.K. Misra, L.E. Misiak, D. Bahadur, V. Srinivas and R.A. Dunlap, *Phys. Rev. B*, to be published.

[12] F. Mehran, K.W.H. Stevens and F. Holtzberg, *Phys. Rev.* **B17**,3702 (1978).

[13] R.B. Giffiths, in Phase Transitions and Critical Phenomena. edited by C. Domb and M.S. Green (Academic, New York) 1972.

[14] A.K. Bandyopadhyay, J. Zaviycki, P. Auric and J. Chappert, *J. Non-Cryst. Solids*, **40**, 353 (1980).

[15] J.J. Hauser, H.S. Chen and J.V. Waszczak, *Phys. Rev.* **B33**, 3577 (1986).

[16] R.A. Dunlap, M.E. McHenry, V. Srinivas, D. Bahadur and R.C. O'Handley, *Phys. Rev.* **B39**, 4808 (1989).

[17] S.K. Misra and K.N. Shrivastava, *Phys. Rev.* B37, 2255 (1988).