DUBLIN CITY UNIVERSITY

School of Electronic Engineering

Master of Engineering

Thesis

THE DEVELOPMENT OF AN INTELLIGENT MOBILE MAINTENANCE WORKSTATION FOR APPLICATION IN AUTOMATED CNC MACHINE REPAIR

Author: Mustafa Houreh B.Eng

Supervisor: Dr. Charles McCorkell

September 1991

THIS THESIS IS BASED ON THE AUTHORS OWN RESEARCH RESULTS

ABSTRACT

The project focuses on electronic repair in CNC machine. This is justified given that electronic causes determine a high percentage of automated CNC machine failures.

An intelligent maintenance workstation is proposed as a vehicle for redressing the problem of uncertainty in the skill level of repair staff and as a means of enhancing the pace of repair under normal circumstances.

The suitability of expert systems in electronic fault diagnosis is established through a review of the literature and a realistic role for an expert system in the definition of the workstation is determined. Circuit interrogation is carried out using a combination of instruments, probes and driving software written in 'C'. Procedural guidance available from the expert system and detailed circuit knowledge is reproducible in the workstation from pre stored CAD data base files. In the absence of a CNC machine to carry out the implementation, a partial implementation only was possible. However all of the elements were included and credibility given to the proposal.

ACKNOWLEDGEMENTS

I am indebted to my project supervisor, Dr. Charles McCorkell, for his continued help and encouragement. Thanks to all the staff of the School of Electronic Engineering at Dublin City University, especially Mr. John Whelan and Mr. Noel Murphy for their help. To my fellow Postgrads and friends at Dublin City University, I wish you good luck with your respective careers.

I must acknowledge the great help given by Mr. John Whelan, in advising me during builting my project.

Finally my sincerest gratitude to my wife for her patience, constant support and understanding throughout.

Declaration

I hereby declare that this thesis is entirely of my own work and has not been submitted as an exercise to any other university.

Mustafa Houreh

DEDICATION

To my parents, as they enter their golden years.

CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Motivation for the Project

During the last 20 years, the complexity of the mechanical aspect of machine tools has significantly increased; this has been accompanied by a corresponding complexity of electronics.

In the early 1970's, Alfred Herbert limited introduced a variety of computer numerical control CNC turning machine tools. These were continuously modified in the light of experience gained in the field; subsequently, updated models were developed and, more recently, microprocessor control, together with manual data input CNC systems were introduced.

Because of its considerable inherent flexibility, a CNC machine is capable of producing an output (machining rate) of up to three to four times the output of manually operated conventional machines. Because of the high output of these machines, breakdowns have an increased significance, since the breakdown of a single CNC machine can result in the production of an entire workshop being halted [34].

In short, CNC equipment availability seldom reached much beyond the 60 percent level, because nine times out of ten the maintenance people could not make the necessary repairs. So it was necessary to call in the machine manufacturer's maintenance people, who, of course, were not immediately available, so the down time was measured in days.

To solve the maintenance problem, a technician was hired, to specifically cope with control unit emergencies and maintenance. So he instituted a preventive maintenance program, and he was able to restore equipment to operating condition within an hour or two (when parts were available) instead of the two or three days. But when this technician left the plant the situation reverted to its original status. The search for a more practical solution led some plants to a nation wide CNC maintenance service, that means they signed a contract with the manufacturer's for minor and major preventive maintenance, which is the key to improve operational efficiency [3].

JACK MOORHEAD [4] recommended the provisions for programming maintenance and operation to be made before the CNC machine tool arrive on the shopfloor, and the CNC users should be in a position to support and sustain CNC operation internally, even though there is considerable assistance available from the outside.

He said, that "to develop this internal capability it is important to select the right people and train them thoroughly.

Three groups are involved :

(1) part programmer.

(2) machine operators.

(3) maintenance technicians."

He added "a minimum number of adequately trained people can keep the CNC running".

Also when these trained people left the plant the situation reverted to its original status, and the breakdown time will cause a halt of the production to the entire workshop.

To solve the problem of depending on expert people, some companies developed expert systems. however, until the early eighties, expert systems required tremendous development work and often special purpose hardware.

"The appearance of micro-computerization and more user friendly computer program languages led us to investigate the application of these new techniques to corrective maintenance of NC machine-tools" [33].

1.2 Summary of the Project

Usually, any workshop, which contains a large number of CNC machines, has its special workstation. This workstation is controlled by a service engineer, who is responsible for solving any problems that might happen to any of the machines in the workshop.

In the case of a breakdown to any machine, the service engineer will collect all the information about the machine, and equiped with service catalogues and some testing devices, will go to the machine location to fix it.

That work may take one or two weeks with a non-expert person, and may only take hours with an expert. As a result, maintenance engineers are forced to keep up with advances in any technology that will reduce downtime.

But as mentioned, in the case of the absence of the expert person, the maintenance will depend on the non-expert, who will take a long time to fix the broken machine.

This project concentrated on CNC machine corrective maintenance. So an intelligent mobile workstation was developed for this purpose. Fig. 1-1, shows the intelligent mobile workstation (mobile unit) inside the CNC machine workshop. This intelligent mobile workstation is controlled by EXP-Test System (the expert system

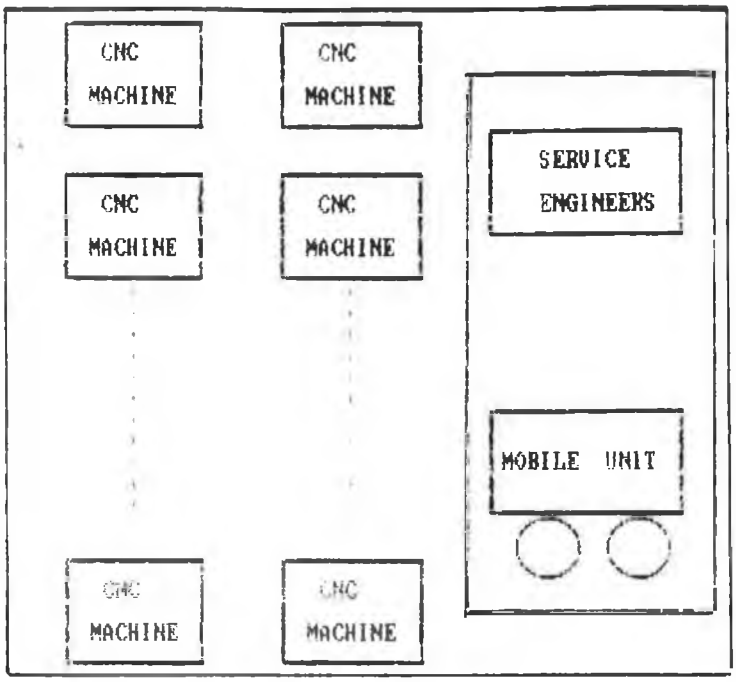


fig. 1-1. The Intelligent Mobile Workstation

which was developed inside VP-Expert system shell). EXP-Test System is a highly flexible, user friendly, and pattern_directed inference system, that is adequate for hardware fault diagnosis, and is accompanied by a two dimensions graphic system; a functional dimension, and a physical dimension. The physical dimension sketches the circuit layout and the functional dimension sketches the circuit schematic. this is important for accessing any component on the circuit.

As a result this will reduce the breakdown time of the CNC machine and will allow any maintenance engineer or technician with little experience to repair the machine quickly. This he can do without returning to the service manual for information about the broken machine, having an integrated system which contains complete information, procedural and graphic.

The intelligent mobile workstation (mobile unit) has five

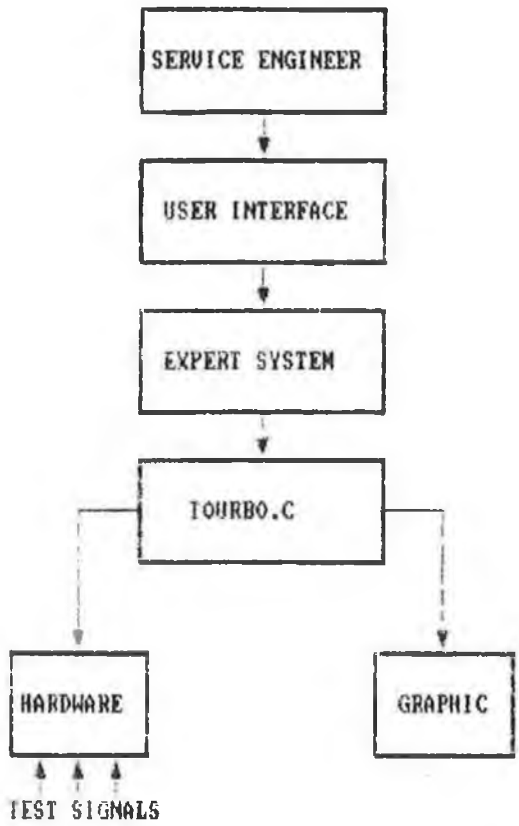


fig. 1-2. The Components of the Mobile Unit

components as shown in fig. 1-2 :

(1) User interface (P.C. screen).

(2) Expert system (VP-EXPERT).

(3) TURBO.C system (TC2).

(4) Graphic system (OrCAD).

(5) Hardware system (controller, IEEE interface card Digital Oscilloscope, and Logic Analyzer).

In the case of a problem the service engineer will move the mobile unit to the location of the broken machine, switch on the unit and follow the instruction of the expert system.

The main input to the system is the test signal from the test equipment. This signal is compared with the reference signal inside TURBO.C system, then the result is chained through the function test rules and a list of components in which the fault might lie is produced, this is the fault candidate list.

The expert system will inform the user and guide him on were to put the probe to get the test signal. In addition it will call a program in TURBO.C, which converts an OrCAD program for drawing the schematic and layout diagrams by (HP plotter), and sketch the mentioned diagrams on the screen of the P.C. using the graphic library in (TC2).

The system backward-chains through these rules with the goal of discovering which of the fault candidates is indeed faulty. If it is difficult to get some information using the test equipment, the Expert System will query the user for that information. So the system gets the extra information that it needs to isolate a fault by asking the user about the faulty device, and the existence and nature of signals at various points in the circuit.

1.3 Aim of the Project

The aim of the project was to develop a mobile intelligent workstation that will:

(1) Include an expert system for electronic fault diagnosis which incorporates both deep and shallow knowledge.

(2) Be independent of the need for an expert service engineer.

(3) Achieve a quick and easy corrective maintenance to any CNC machine.

(4) Decrease the number of service engineers and technicians that are needed for maintenance.

(5) Decrease the downtime of any CNC equipment.

(6) Transfer the knowledge between service engineers.

CHAPTER 2

WORKSTATION SPECIFICATION

2.1 Introduction

As mentioned in the previous chapter, a CNC machine has high output, which increases the significance of breakdown time since the breakdown of a single CNC machine can result in the production of an entire workshop being halted. The maintenance workstation associated with a CNC workshop has more significance than a workstation inside other workshops where linked production is not going on. This chapter will describe the CNC machine maintenance workstation specification and the software tools which are designed to reduce breakdown time.

2.2 General Workstation Specification

In general every workstation will have a number of specifications, to achieve its aims.

In this case they can be broken down into:

1. Service engineer specifications.
2. Test equipment specifications.

2.2.1 Service Engineer Specifications

For the propose of maintaining the CNC machine, which is a very complex machine, an expert service engineer who will have worked in the maintenance field for a long time, following a training course at a CNC machine manufacturer is necessary.

So the service engineer would typically have as minimum specification:

1. A good knowledge in maintenance from experience.
2. A specific background in fixing CNC machines.
3. Because circuit complexity is increasing (doubling about every two years) he must be able to keep up with advances in any technology that will reduce downtime.

2.2.2 Test Equipment Specification

Usually in any workshop which deals with manually controlled machines, the basic test tools which are adequate for testing power supply, some electrical components, and simple logic circuits should be available. These tools include :

1. Analog multimeter.
2. Digital multimeter.
3. Oscilloscope.
4. Function generator.
5. Power supply.

The test or service equipment used in CNC machine troubleshooting is basically the same as that used in other fields of electronics. That is, most procedures are performed using conventional meters (including high voltage meters for measurement of video terminal CRT voltages), multitrace oscilloscopes (for measurement of pulses on data and address buses, clock and other control lines, etc.), and assorted clips, patchcords, power supplies, and hand tools. Theoretically, all CNC machine troubleshooting problems can be solved using such instruments.

However, there are some specialized test instruments that greatly simplify microcomputer service (just as they do for any programed, digital device composed mostly of ICs). such specialized equipment includes :

(1) LOGIC PROBE

Such probes will detect and indicate high and low (1 or 0) logic levels, as well as intermediate or "bad" logic levels, including an open circuit, on a single line of CNC machine circuit.

(2) LOGIC PULSER

Such pulsers are hand-held logic generators used for injecting controlled pulses into digital logic circuits.

(3) CURRENT TRACER

Such current tracers are hand-held probes that enable the precise localization of low-impedance faults in electrical systems (including typical CNC machine printed-circuit wiring).

(4) LOGIC CLIP

Such clips are designed for logic-level determination only on ICs using TTL and DTL circuits.

Generally, clips can test flip-flops, gates, counters, buffers, adders, shift registers, and the like, but will not test ICs with nonstandard input levels or expandable gates.

(5) LOGIC COMPARATOR

Such comparators clip onto 16-pin ICs and, through a comparison scheme, instantly display any logic state differences between the test IC and a reference IC.

Logic differences are identified to the specific pin or pins of the IC with the comparator's display of 16 LEDs. A lighted LED corresponds to a logic difference.

(6) LOGIC ANALYZER

Although single stepping and a check of system timing can pinpoint many CNC machine problems, there are two obvious problems. First, present-day multitrace oscilloscopes do not have enough traces to accommodate all the lines in data and address buses simultaneously. Second, and more important, a typical data byte is 8 bits, and thus requires 8 clock pulses, or 8 one-at-a-time pushes of the single-step button. Since all program steps require at least one byte (and often two or three bytes, possibly 24 bits), you must push that button many times if the malfunction occurs at step 3333 of the program! This means that you must spend endless hours comparing program lists against binary readouts at addresses.

(If you are already familiar with troubleshooting of any programmed device, you know that the most time-consuming part of the task is in making such comparison.)

The logic analyzer overcomes this basic problem by permitting you to select the data at a particular address for display. The logic analyzer will then run through the program at near the normal system speed (a fraction of a second) and display the selected data at the desired breakpoint or between two breakpoints in the program [69].

2.3 Breakdown Time

In the case of a breakdown to any CNC machine in the mentioned workshop, the service engineer will collect all the information about the broken machine and equipped with the service catalogues, and all the test equipment which have just been mentioned, will go to the machine location to fix it.

That work may take one or two weeks with a non-expert person (service engineer or technician, who doesn't meet the previous specification for the service engineer sec. 2.2.1), and may only take hours with an expert (person who has the service engineer's specification).

2.4 The Alternative Solution

To solve the problems which will face any manufacturer when installing workstations inside his plant, which contain CNC machines, and to allow service engineers or technicians to carry out the necessary maintenance to any of the machines, an intelligent mobile unit was developed, controlled by an expert system.

Two situations have been mentioned viz:

1. Maintenance where the skill level of the maintenance engineer or technician is uncertain.
2. Maintenance where the skill level of the maintenance engineer is high.

In specifying the workstation, the intention is to assume the realistic situation of an uncertain skill level in the maintenance staff (arising through inadequate training, turnover etc.) whilst attempting through maximum support in the workstation, to achieve the level of performance possible with the highly skilled maintenance person.

Translating this intention into appropriate technology-hardware and software- is the essence of this project.

The make-up of the hardware has already been hinted at. The major components are:

(1) A controller (Personal Computer).

(2) Digital Oscilloscope.

(3) Logic Analyzer.

(4) A suitable hardware interface.

(5) Probes and other ancillary equipment.

In chapter three the hardware make-up of the workstation is dealt with in more detail.

In defining the software many more issues arise such as:

1. What routine software is necessary to drive the instruments?
2. How should the man_machine interface be defined?
3. What level of support is possible with an expert system?
4. If an expert system is to be used, can an off-the-shelf expert system be integrated with a standard programing language such as 'C'?
5. How do we define the knowledge base given the context- electronic circuit knowledge is usually documented graphically yet maintenance procedures may be stated linguistically?
6. Maintenance procedures and solutions can be at two very distinct levels- shallow and deep- how do we accommodate that?

These are examples of the many unfolding issues that arise in any attempt to define the software. Most of the agonizing centres around the question of knowledge encapsulation and representation. Following considerable reading in this area we concluded that any attempt to encapsulate known diagnostic procedures in a linguistic based expert system would be impossible given the near infinity of loops and breakdown possibilities in even modestly complex electronic and electromechanical systems. The partitioning of knowledge into "shallow" and "deep" is useful and has been used as follows.

Routine isolated faults that constitute the largest group of all faults (power supply problems for example) are classifiable as shallow from a knowledge point of view.

Any problem which requires information with detailed circuit information which is normally preserved graphically is classified as 'deep' from a knowledge point of view.

The following assumptions underpin the proposed intelligent maintenance workstation.

1. Shallow knowledge with diagnostic procedures can be encapsulated in a standard expert system shell and will enhance the pace of early fault diagnosis.
2. The normal context applying in the diagnosis of complex faults must be acknowledged in the sense that:
   (a) Graphical information is available and used.
   (b) convergence to fault identification is open-ended, uncertain, interactive, involves learning whilst persueing the fault, and the outcome is unpredictable.

The conclusion reached in defining the software specification therefore is that all of the following elements must be available.

1. Routine software for signal detection and instrument driving.
2. An expert system shell for shallow knowledge encapsulation and user guidance.
3. A graphical facility to allow the introduction of "deep knowledge" in the context of complex fault diagnosis.

Specifically the following hardware and software elements constitute the workstation.

This unit consists of four hardware parts which are:

(1) Controller (personal computer).

(2) IEEE interface card.

(3) Digital oscilloscope.

(4) Logic analyzer.

And from four software packages which are:

(1) VP-EXPERT system.

(2) TURBO.C system.

(3) Orcad system.

(4) IEEE software system.

By including all the information which is important for fixing CNC machines, plus the schematic diagram and layout circuit diagram, inside the data base of the expert system, and by making the data base follow the maintenance tree, the CNC machine maintenance can be made very easy.

As a result this will reduce the breakdown time of the CNC machine and will allow any maintenance engineer, or technician with limited experience of repair of CNC machines to complete the service without having to refer to paper based information.

CHAPTER 3

HARDWARE CONFIGURATION

3.1. Introduction

As mentioned in chapter 1, the intelligent mobile unit has five components, which are :

(1) User Interface.

(2) Expert System.

(3) TURBO C. System.

(4) Graphic System.

(5) Hardware System.

This chapter will give a more specific discussion about the hardware system, which is illustrated in fig. 3-1 and consists of five parts:

(1) Controller (P.C.).

(2) IEEE Interface Card.

(3) Digital Oscilloscope.

(4) Logic Analyzer.

(5) Cables.

3.2. The Controller

The controller sends the command signals to the test equipment (Digital Oscilloscope and Logic Analyzer), through the IEEE card, those command signals cause :

1. Initialize the IEEE bus and the instruments.
2. Sending the setup information to the test equipments.
3. Command the instruments to get the information from the

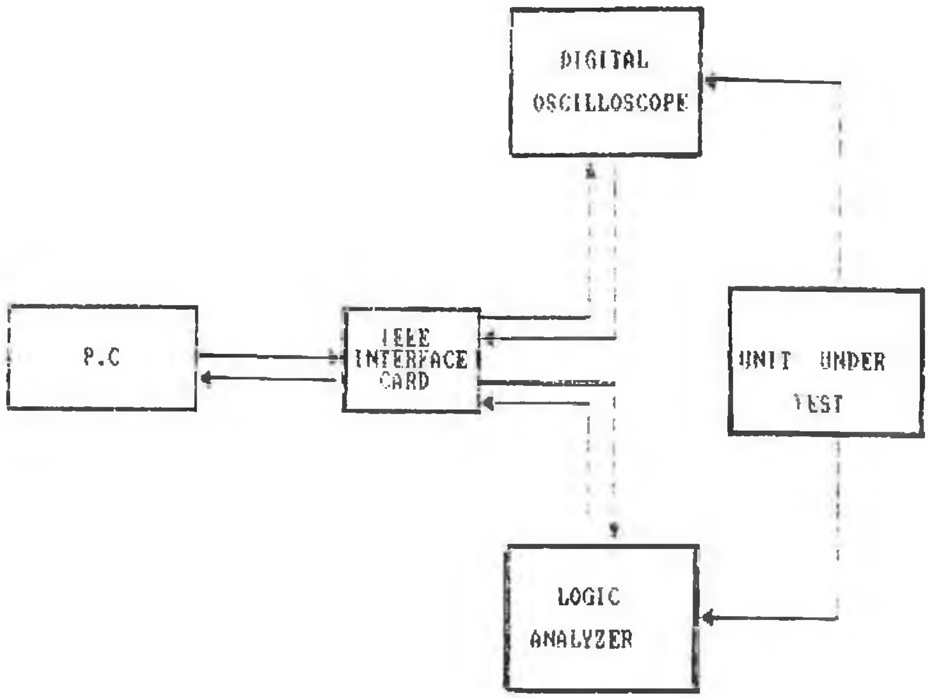


fig. 3-1. The Hardware Configuration

U.U.T..

4. Command the instruments to send data to the controller.

5. Check that the correct information was received.

After the controller receives the information from the instruments, it compares that information with the reference.

3.3. IEEE Interface Card

The intention is to look into automating a test procedure for a volume product, so as to reduce costs. This calls for some kind of computer, a program and some interface to particular pieces of test equipment needed for this job.

Unfortunately, each piece of test equipment has a different number and type of control input, this will require a number of different interface cards in the computer, each with a different software driver.

All this interfacing is an overhead to actually doing the real job of specifying and writing the program that manipulates the test instruments to automate the test procedure.

What is required, is that every test instrument have the same interface cards. It would be preferable if communication via a common bus into one card in the computer were available.

Before the solution is described, a brief review of Digital Communication principle is given.

3.3.1 Introduction to Digital Communication

Serial and parallel data transfer are the two basic methods of communicating digital information between microprocessors and peripheral equipment. Both techniques are in widespread use throughout the industry and each has its advantages and disadvantages.

3.3.1.1 Serial Data Transfer

Serial data transfer is commonly used whenever digital information must be relayed over a relatively long distance.

The data is often transferred through the telephone wires or over the airwaves via some form of radio carrier.

The main reason for long-distance serial transfer is the reduction in the number of wires required to carry the information. Unfortunately the speed at which this data can be transferred serially is normally limited to, at present, no more than 4,800 bits per second over commercial voice grade telephone equipment. Leased service is available for rates of 9,600 bits per second.

In theory a voice grade channel can carry up to about 20,000 bits per second. Higher speeds are attainable if special digital communications links are leased from the telephone company.

3.3.1.2. Parallel Data Transfer

Parallel transmission is used for short distances where the speed of information transfer is critical. This form of data communication is found in newer types of computer peripheral equipment with transfer speeds of up to one million characters per second. This equipment includes printers, disk drivers and various other forms of peripheral components.

3.3.1.3 Asynchronous Serial Data

Serial data is transferred in either the asynchronous or synchronous form. In asynchronous transmission, some times referred to as start-stop transmission, start and stop bit intervals are transmitted with each byte of information for the purpose of synchronization. No clock waveform is transmitted with asynchronous data, since the start and stop bits are used for synchronization.

In synchronous data transmission, synchronization is effected by transmitting a synchronization character or two, followed by a large block of data. In addition to the sync characters, a clock waveform must also be transmitted.

Therefore, synchronization occurs for a block of data in a synchronous system and for each piece of data in an asynchronous system.

Fig. 3-2 illustrates the typical format used for transmitting data asynchronously. Each piece of information is preceded by a start bit that is at a logic zero or, by definition, a space. This is

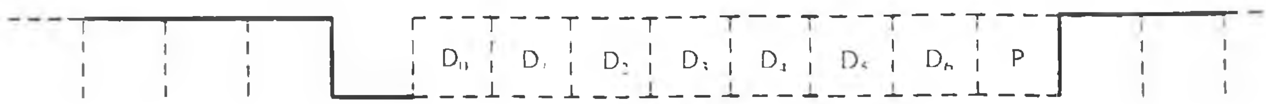


fig. 3-2. Asynchronous Serial Data

followed by data bits that comprise the information that is always transmitted with the least significant bit first. The stop bit, or bits in some older systems, follows the data and is always at the logic one level or, by definition, a mark.

3.3.1.4. Baud Rate

The speed at which serial data is transferred is referred to as its baud rate. The baud rate is arrived at by taking the reciprocal of the bit time interval for most application. Refer to the section on PSK (phase shift keying) for a different definition of baud rate as it applies to that form of data. For example, a bit time of 9.09 ms would have a rate of 110 baud, except for PSK. If the serial message consists of a start bit, 8 data bits , and 2 stop bits, a system working at this rate would be capable of transferring 10 bytes of data per second.

Table 3-1 illustrates some commonly used baud rates, along with the number of stop bits and data bits, type of transmission, and the normal application of each.

Note that all of the baud rates listed are multiples, except 110 baud, which is used for communications between electromechanical teletypewriters (which are quickly disappearing). The only systems employing 2 stop bits or 1.5 stop bits were designed for mechanical devices. The extra time allowed by additional stops was required

Baud	Data Bits	Stop Bits	Type	Application
110	5	1.5	Asynchronous	Baudot TTY
110	7 + P*	2	Asynchronous	ASCII TTY
300	7 + P	1	Asynchronous	FSK MODEM
600	7 + P	1	Asynchronous	FSK MODEM
1200	7 + P	1	Asynchronous	FSK MODEM
2400	Variable	–	Synchronous	PSK MODEM
4800	Variable	–	Synchronous	PSK MODEM
9600	Variable	–	Synchronous	PSK MODEM

NOTE: *P = Parity TTY = Teletypewriter
MODEM = MOdulator DEModulator

table. 3-1. Commonly Used Baud Rates

for mechanical synchronization in these devices. All other systems use 1 stop bit.

3.3.1.5 Synchronous Serial Data

In synchronous transmission, data is transmitted with clock pulses, so it is not necessary to send synchronization bits along with the data, as with the asynchronous system.

Synchronization can be accomplished by transmitting sync information periodically.

For example, transferring 100 bytes of information by asynchronous methods would take 1000 bit times. This assumes that 1 start and 1 stop bit per byte of data is being transmitted. In a synchronous system that sends 1 sync byte before the start of transmission and end of message character at the end of transmission, it requires only 816 bit times.

If information is transmitted for any extended period of time, synchronous communication is obviously much more efficient.

Since this synchronous communication can take many forms; many computer manufacturers have developed standard communications protocols, such as BISYNC (binary synchronous communications), and HDLC (high-level data link control).

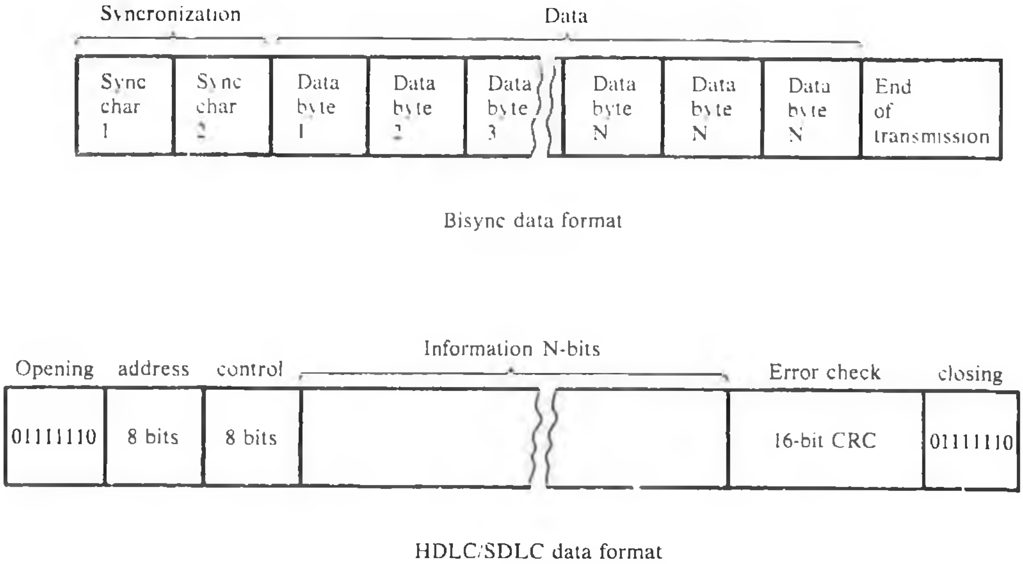


fig. 3-3. Two Forms of Synchronous Serial Communications Protocols

These type of protocols are illustrated in fig. 3-3. BISYNC is a byte oriented protocol; SDLC and HDLC are both bit oriented protocols [76].

3.3.2. The Solution

The IEEE-488 parallel data transfer standard was developed by the Hewlett-Packard Corporation for use in their instruments. It was adopted as a general purpose instrumentation bus (GPIB) by IEEE in 1975. This standard is like the RS-232C standard because it defines the pin connections, protocol, and standard messages for communications.

The IEEE-488 interface bus is a means of connecting a number of instruments to a common input/output port of a computer.

An 8 line bus carries addresses, data and commands between instruments and the computer. Each instrument has an address or addresses assigned to it (usually by means of a DIL switch on the instrument) and can be called up by the computer.

After activation the addressed instrument may either send data (talk) or receive data (listen), before being de-activated (untalk, unlisten) and the next instrument activated.

An instrument that can send data is called a talker. Likewise, one that can receive data is a listener and one that can send or receive is called a talker, listener. The computer is called the controller.

There may be only one controller active on the bus at any time, although control may be transferred from one controller to another. Also, there may be only one talker active on the bus at any one time, although there may be many listeners if desired.

Physical (electrical) limitations are, up to 15 devices connected to the bus, with no more than 2 meters between them or 20 meters bus length overall. Also, more than 50% of the instruments connected must be powered up even if not used.

nowadays, bus extenders exist to both extend the bus in length and to increase the number of instruments on the bus.

3.3.3. The IEEE-488 Bus Lines

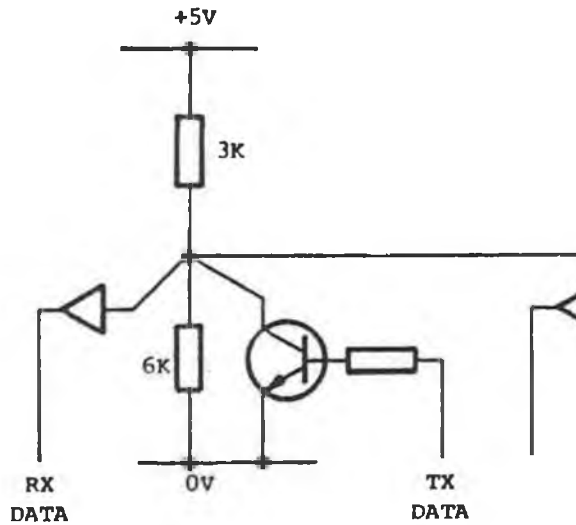
Referring to fig. 3-4 the bus comprises a number of lines connected to each instrument in turn.

The electrical nature of the interface at each instrument is show in this figure. As a "wire OR" function is implemented, the bus line is active or true when low, i.e. at 0.4v or less.

When nothing is driving that particular bus line, it will sit at about +3.3v and be read as inactive or false.

The maximum bus transfer speed is supposed to be one million transfers per second. In actual fact, very few measurement set_ups

THE IEEE 488 BUS LINES

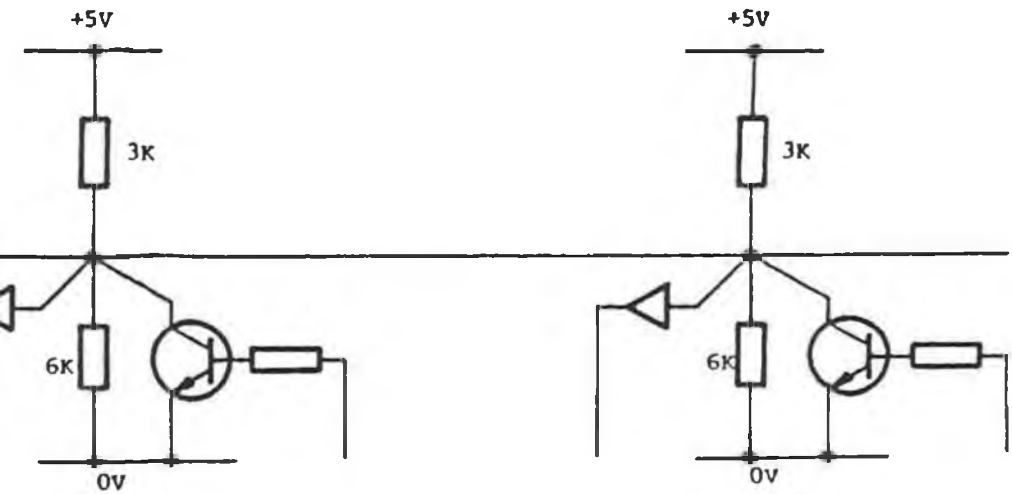


CONTROLLER

BUS LINE "INACTIVE" V_B = +3.3V

BUS LINE "ACTIVE" V_B = +0.4V

fig. 3-4



INSTRUMENT 1

INSTRUMENT 2

come anywhere near this speed.

Referring to fig. 3-5 the bus is made up of 16 lines and 8 grounds making 24 lines in all.

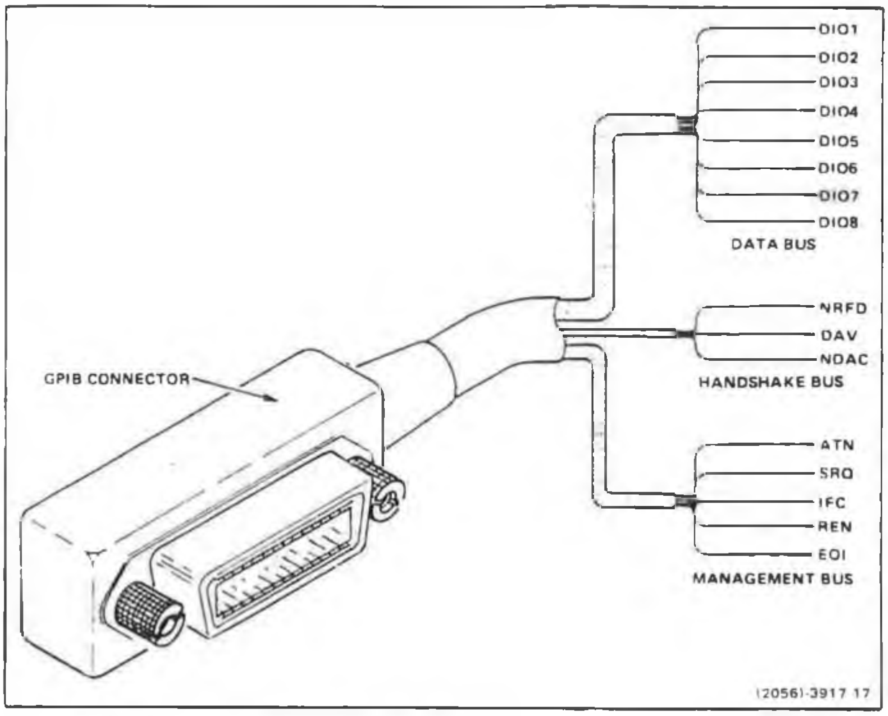


fig. 3-5. The GPIB Bus Lines

Ignoring the grounds for the meantime, the 16 lines are divided into three groups, named transfer control lines or Handshake, Bus Management lines and the Data Bus.

The Data Bus consists of eight lines carrying all commands, data and addresses bi-directionally between instruments and controller.

The Handshake lines are manipulated to ensure that each 8 bit byte is successfully read by the listener.

The Bus Management lines are primarily used to differentiate between data, commands and addresses on the bus, as well as a number of miscellaneous control functions regarding the passing of data, status and command streams between devices.

3.3.3.1. Data Connections

The data connections may be open collector or three-state logic. In most applications these lines use three-state bidirectional data transceivers, since they increase the usable frequency range of the bus. The frequency range is increased because a three-state driver contains a low impedance pullup network that charges the line capacitance more quickly. This reduction in zero to one transition time allows higher data transmission rates. The data pins are labeled DIO1 through DIO8, where DIO1 is the least significant bit position [76]. The actual pin numbers are illustrated in fig. 3-6.

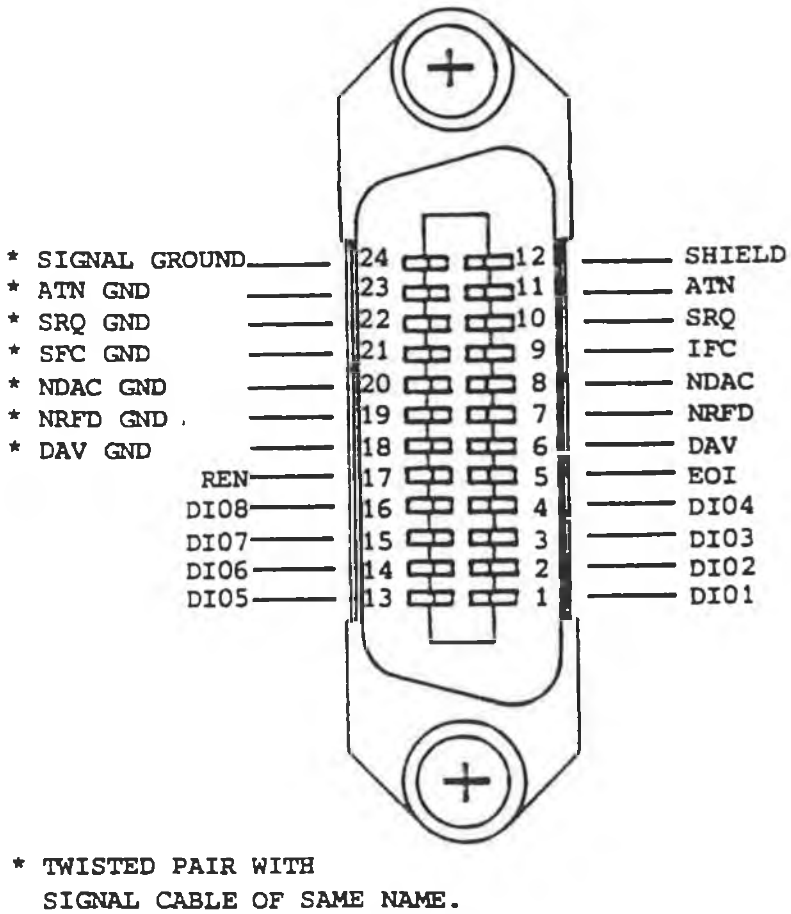


fig. 3-6. GPIB Interface Connector Plug

3.3.3.2. Handshaking Connections

To ensure that bytes pass from talker to listener(s), a three line handshake is implemented with DAV, NDAC and NRFD.

DAV The DAV, or data available, is issued by the talker to indicates the availability or validity of data on the data bus connections.

NRFD The NRFD connection, or not ready for data pin, is issued by the listener to indicates the readiness of the device or devices connected to the bus to receive data.

NDAC The NDAC, or not data accepted, is issued by the listener to indicates the condition of acceptance by the devices connected to the bus.

This handshake allows bytes to be passed at the rate that the slowest listener can take them. The handshake occurs for every byte transfer, irrespective of whether that transfer is data, commands or addresses.

Before a byte is sent from the talker to the listeners, the three handshake lines should be in the states given by table 3-2.

LINE	STATE	MEANING
DAV	High	Data not available
NRFD	High	Ready for data
NDAC	Low	Data not accepted

table 3-2. Quiescent State of the Handshake Lines

Referring to fig. 3-7 when the talker is ready to output data, it inputs NRFD and NDAC (A,B) and, providing they are in the state given in table 3-2, it outputs a byte onto the data bus. After allowing a few nanoseconds settling time it drives DAV low (C) signalling that data is valid on the bus.

Meanwhile, the listeners are looking at DAV, waiting for it to go low and, when it does, they take low the NRFD line (D), signalling to the talker that it must not change it's data.

At the same time, the listeners latch in the data byte and, having done so, release the NDAC line (E) signalling that they have accepted data. Due to the party line nature of the bus, NDAC will not go high until the slowest listener has accepted the data.

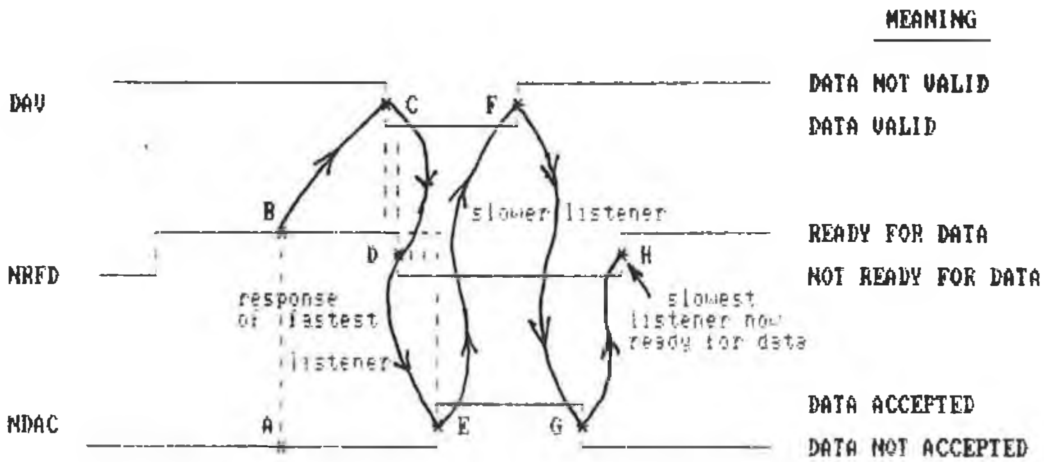


fig. 3-7. The Handshake

Once the NDAC goes high, the talker initiates a reset sequence to prepare for the next byte transfer, if any. This happens by it taking DAV high again (F) and removing data from the bus. The listeners in response to DAV going high, let NDAC go low (G), signalling that they have not yet accepted the subsequent data byte (because it hasn't yet been sent!!) and they let NRFD go high (H), to signal to the talker that they are ready for (new) data.

3.3.3.3 Interface Management Connections

These connections, IFC, ATN, SRQ, REN, and EOI, manage the flow of information through the GPIB.

IFC The IFC, or interface clear, is issued by the controller to initialize all instruments on the bus to their known states.

ATN The ATN, or attention, is issued only by the controller to indicate that the byte on the bus is a command, or address and not data.

SRQ The SRQ, or service request, is issued by any device needing service from the controller.

REN The remote enable connections, is issued by the controller when it wants remote control over all the instruments on the bus.

EOI The EOI, or end or identify, send by the talker with the last byte of a data stream to indicate end of data.

3.3.4. Addressing

As all instruments share a common bus, there must be a method of selecting an instrument from others to make it do something specific.

The IEEE-488 bus protocol does this by allowing the controller to send an address down the bus before some command or data is sent. The addressed instrument recognizes its address and prepares itself for some kind of activity.

An address may be sent down the bus by making ATN low, at the same time as transmitting a Hex address in the range 20 to 3E (corresponding to address 0 to 30 in decimal), if the instrument is to become a listener. Alternatively, making ATN low with Hex 40 to 5E will make the addressed instrument a talker.

Instruments generally have the same address for talk or listen as set by the address selecting DIL switch, or by the other ways, depending on the instrument it self.
For example, selecting 04 on the DIL switch will cause the instrument to respond to address Hex 24 for listen or Hex 44 for talk.
It should be remembered that more than one listener can be active on the bus at any one time, but not more than one talker.

3.3.5. Setting the DIL switch on HP-IB Interface Card

The configuration switches on the HP-IB interface card set the interface's operating parameters.
They are set at the factory as shown below fig. 3-8.

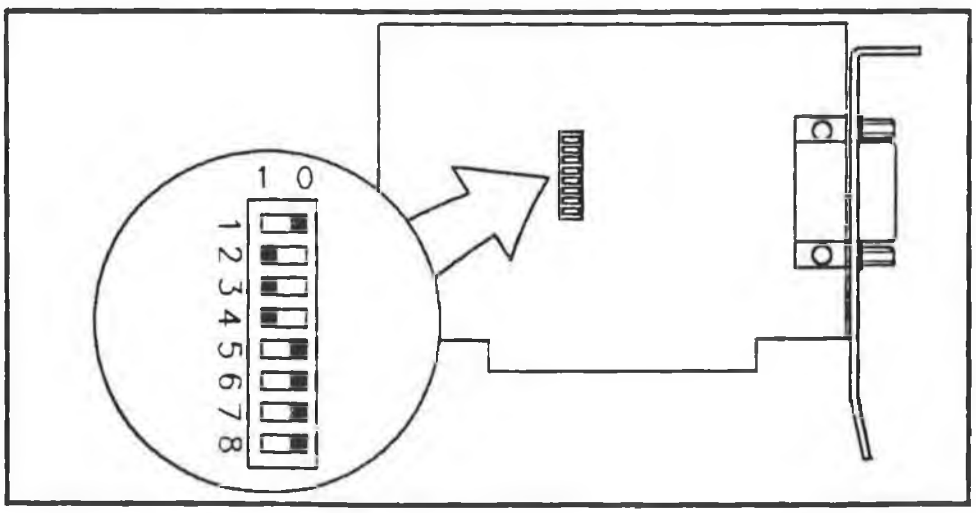


fig. 3-8. The Configuration Switches

This setting specifies select code 7 (memory address DC000) and interrupt level 3.
As a procedure to check the HP-IB interface card configuration, a comparison has been made between the switch setting of this card and with those shown above. As this configuration is satisfactory

for most systems, it was kept as it is.

The address switches (1 through 4) determine the memory address and select code of the interface. If it is wished to change the recommended setting, an address (or select code) that isn't the same as the address (or select code) of any other card installed in the system should be chosen.

Table 3-3 help for set the address and select code, it shows addresses used by some common system components.

Memory Address	Select Code	Switches 1	2	3	4	Potential Conflicts
C0000	16	0	0	0	0	Used by VGA and EGA.
C4000	1	0	0	0	1	Used by VGA.
C8000	2	0	0	1	0	Used by hard-disk controller on pre-1985 Vectra PC.
CC000	3	0	0	1	1	
D0000	4	0	1	0	0	
D4000	5	0	1	0	1	
D8000	6	0	1	1	0	
DC000	**7**	**0**	**1**	**1**	**1**	**Recommended setting.***
E0000	8	1	0	0	0	Used by VGA performance setting on Vectra ES.
E4000	9	1	0	0	1	Used by VGA performance setting on Vectra ES.
E8000	10	1	0	1	0	Used by VGA performance setting on Vectra ES.
EC000	11	1	0	1	1	Used by VGA performance setting on Vectra ES.
F0000	12	1	1	0	0	Reserved for system ROM.
F4000	13	1	1	0	1	Reserved for system ROM.
F8000	14	1	1	1	0	Reserved for system ROM.
FC000	15	1	1	1	1	Reserved for system ROM.

* If you have more than one HP-IB interface, only one can use this setting.

table 3-3. The Address and Select Code

The interrupt switches (5 and 6) determine the level at which the HP-IB Interface interrupts the CPU. Interrupt level 3 is the factory setting.

3.3.6. Connecting Peripherals

An HP-IB system can accommodate up to 14 peripherals in addition to the computer, which is the "controller" of the system. The peripherals have been connected using (RS 489-368) cables 1 meter long. That was done by follow the following instructions:

1. As referred in the peripheral manual to find out how to set up the peripheral, its power cable was connected , and it was switched on (before connecting it to the computer).
2. Then it was turned off.
3. A bus address for the peripheral that does not conflict with the addresses of other peripherals was determined.

   Each peripheral must have a unique address (the HP-IB Command Library uses address 30 for the controller address, so it should not be used).
4. After that the peripherals were connected.

Here are some points which are important when connecting several peripherals to the interface:

(1) Up to 14 peripherals can be connected to a single interface.

(2) Devices can be interconnected in any scheme as long as there is an unbroken path between each peripheral and the controller. Several devices can be connected at one connector, or each device can be connected to the next.

(3) The total cable length on one interface should not exceed 2 meters (6 feet) times the number of connected devices (the computer is considered as one device) and it should not exceed 20 meters (66 feet).

For high-speed data transfer, the length should not exceed 1 meter (3 feet) times the number of devices, and not more than 15 meters (50 feet).

3.4. The Digital Oscilloscope

In this project the HP 54501A Oscilloscope was used as a Digital Oscilloscope, because it is a general purpose, digitizing oscilloscope that is fully programmable and transportable. It is an excellent general purpose digitizing oscilloscope because of the friendly user interface, yet it has many sophisticated capabilities and multiple triggering functions.

3.4.1. Addressing HP 54501A Oscilloscope

The front panel of the (HP 54501A) is separated into six functional areas, as shown in fig. 3-9.

The Menus Section consists of nine keys to select from:

* Timebase
* Channel
* Trigger
* Display
* Waveform Math
* Waveform Save
* Define Measure
* Utilities
* Delta-t/Delta-v

The Display Section contains the screen and selection keys. In a vertical column on the right side of the screen is the function display.

The functions that are displayed at any one time will correspond to a key in the selection column. These keys can select any available function or field that is displayed in halfright.

The Utilities Menu Key should be pressed to address the HP 54501A. This will give access to the calibration and service functions,

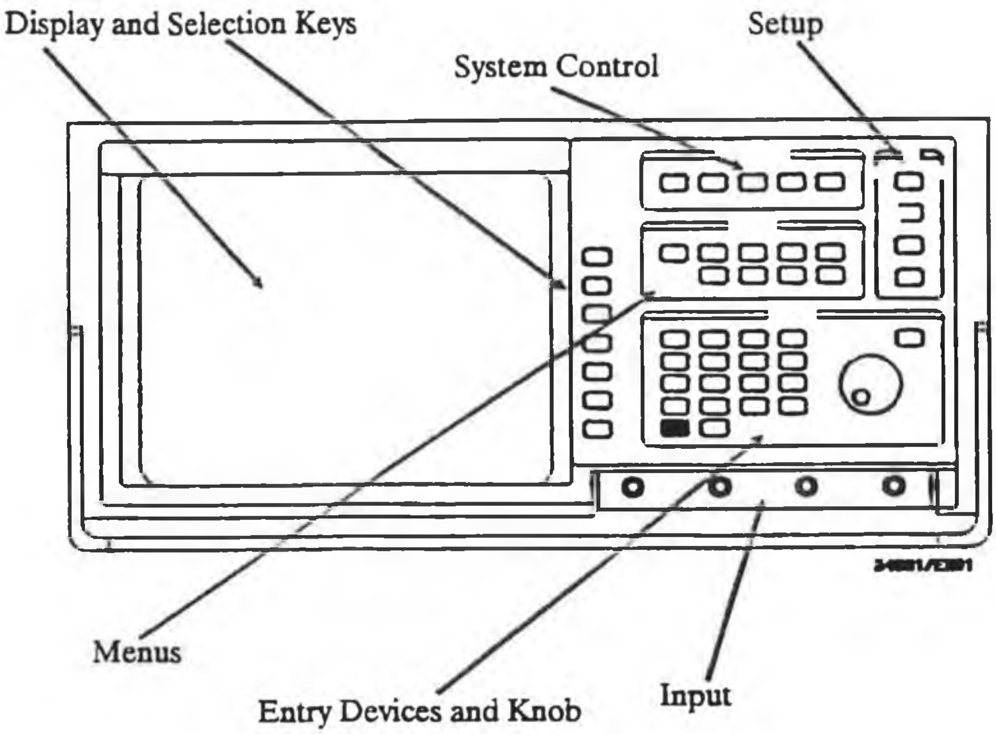


fig. 3-9. The Front Panel of the (HP 54501A)

as well as setup the HP-IB Interface. The submenus which appear at this time includes: (as illustrated in fig. 3-10)

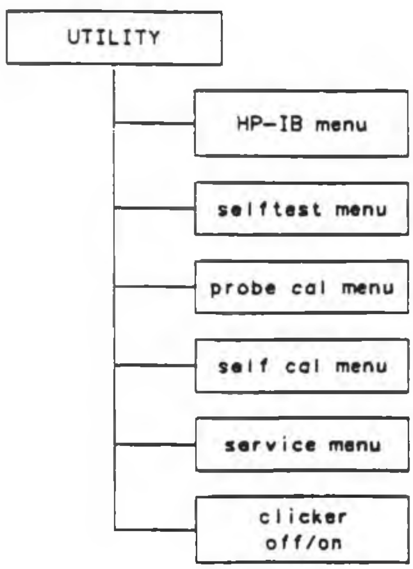


fig. 3-10. The Submenus of the Utilities Menu

* HP-IB menu

* Selftest menu

* Probe cal menu

* Self cal menu

* Service menu

So if the selection key which matches the HP-IB submenu is pressed, it will allow the user to make settings so the HP 54501A can talk to peripheral devices.

As illustrated in fig. 3-11, this interface includes:

* Talk only mode

* Addressed mode

* EOI

* Form feed

* Paper length

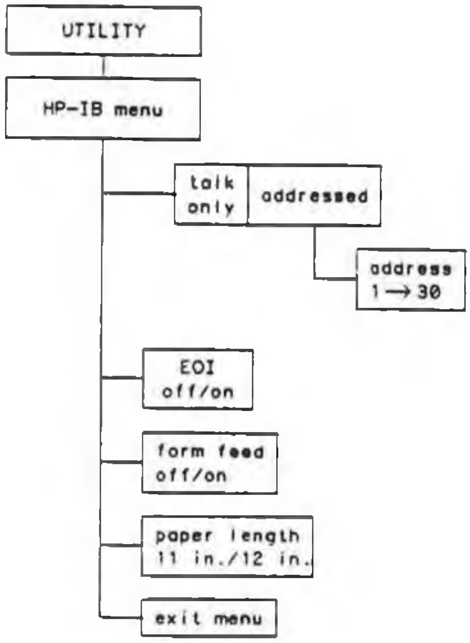


fig. 3-11. The HP-IB submenu

1. Talk only mode

Set the oscilloscope to talk only when a hardcopy, without intervention from an external controller, is wished. The attached printer must be set in the listen only or listen always mode.

2. Addressed mode

   This mode allows a controlling device to selectively address the HP 54501A for talking or listening. The address of the HP 54501A can be selected while the instrument is in the addressed mode.

3. EOI key

   The EOI (End or Identify) key toggles this function on or off. EOI is a line on the HP-IB asserted with the last data byte of a message.

   If this function is on, EOI will be asserted by the HP 54501A on the last byte of each message sent. If it is off, EOI will not be asserted.

   This function only affects messages sent from the HP 54501A. The HP-IB will accept any of the legal IEEE-488 message terminators regardless of the setting of this function.

   IEEE-488 requires that EOI is asserted. Therefore, with EOI off, the HP 54501A will send messages that do not follow IEEE-488 rules concerning EOI. EOI should only be turned off if the controller does not deal with EOI appropriately.

4. Form Feed key and Paper Length key

   Are useful when the oscilloscope is connected to a printer.

In this project the addressed mode was selected, then address (7) was given to the oscilloscope and case "on" for the EOI was selected (as illustrated in fig. 3-12)

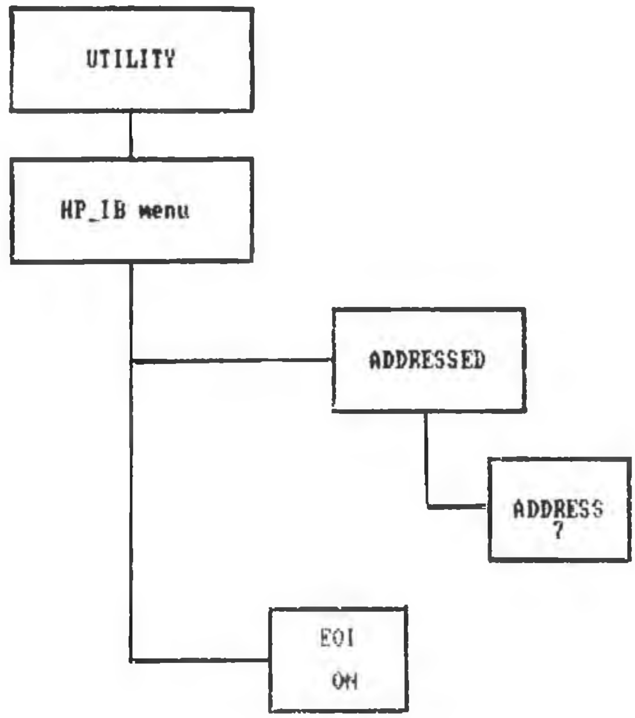


fig. 3-12. Addressing HP 54501A Oscilloscope

3.5. The Logic Analyzer

The Tektronix 1241 was used in this project as a logic Analyzer, because it is a portable, general purpose, digital design and troubleshooting tools that offers similar feature sets. The 1241 Logic Analyzer uses a liquid crystal color shutter (LCCS) to produce a three color display screen, it also has a vertical expansion feature that doubles the height of the timing diagram traces.

Two types of data acquisition cards allow the instrument to be configured to meet the specific requirements. The 1240D1 card supports high-speed hardware analysis with 9 acquisition channels at 100 MHZ (10 ns) and 6 ns glitch detection. The 1240D2 has 18 acquisition channels at 50 MHZ and includes a bus demultiplexing feature.

Instrument configurations include any combination of 1240D1 and 1240D2 acquisition cards up to a maximum of four cards. A 1241 configured with both card types is an effective tool for evaluating hardware-software integration.

The 1241 Logic Analyzer provide the following features:

(1) Acquisition with one or two timebases. Asynchronous or synchronous selections are available.

(2) Powerful triggering with two event recognizers that can be used independently or together. The global event recognizer triggers on a single event in one or two timebases. The sequential event recognizer consist of up to 14 separate levels. Each level specifies its own event and a trigger, wait, jump, delay, or reset action.

(3) Data display in state table or timing diagram formats.

(4) Simple, menu-oriented user interface featuring a front-panel keyboard and a display screen with touch_sensitive, on-screen softkeys.

(5) Auto-acquisition mode, for repeated acquisitions without manually restarting the instrument.

(6) Expandable acquisition memory depth with a memory chaining feature.

(7) Data search and compare functions.

(8) Non volatile memory ensures that two instrument setups (the current setup and another setup of the user choice) are not lost when power is turned off.

(9) Full programmable (by adding the GPIB COMM pack to the Logic Analyzer.

3.5.1. Addressing 1241 Logic Analyzer

The operations of the 1241 Logic Analyzer are controlled through menus displayed on the screen. Use the MENU keys on the front panel to select general menu groups, select individual menus through on-screen soft keys. Each soft key is outlined on the screen and has a label describing its function.

The menu groups as illustrated in fig. 3-13, consist of five keys to select from:

* CONFIG
* TRIGGER
* DATA
* EDIT
* UTILITY

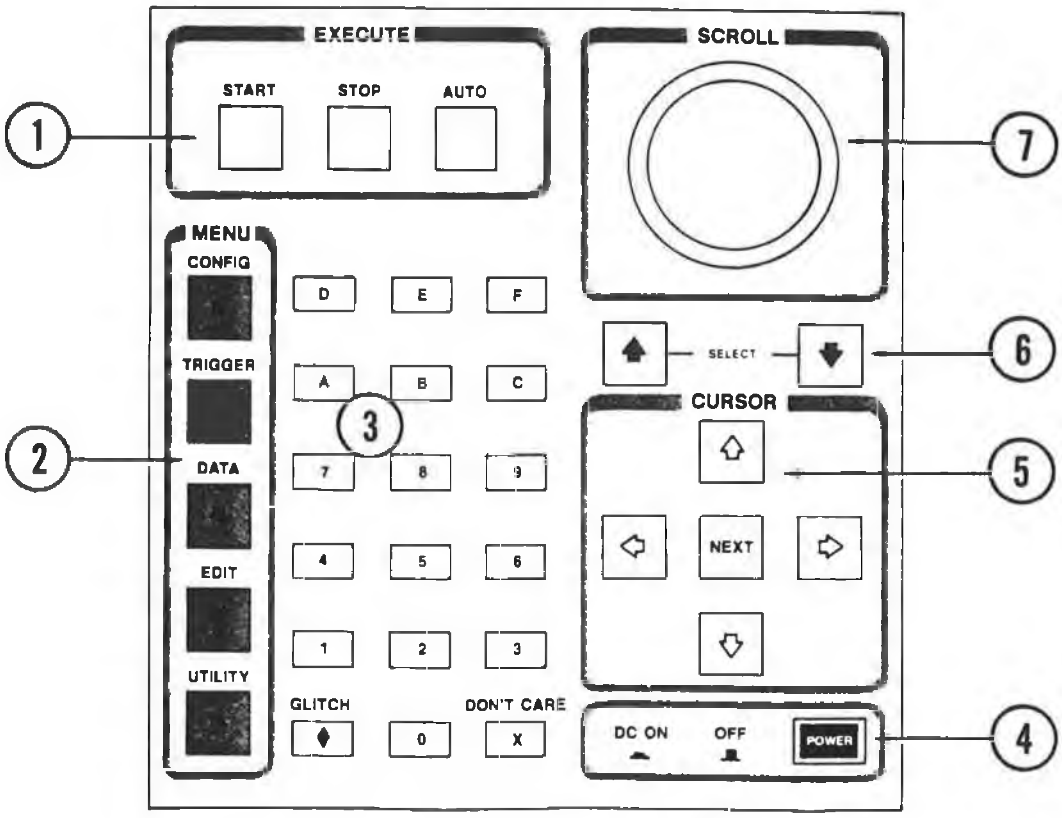


fig. 3-13. The Front Panel of the 1241 Logic Analyzer

The UTILITY menu key should be pressed to address the 1241 Logic Analyzer. This will give access to the STORAGE MEMORY MANAGER menu, a COMM PORT CONTROL menu (if COMM Pack is installed) and ROM Pack menu (if ROM Pack is installed).

So once the COMM Pack has been installed, (as shown in fig. 3-14), enter the menu by pressing the UTILITY key, and then the COMM PORT CONTROL soft key.

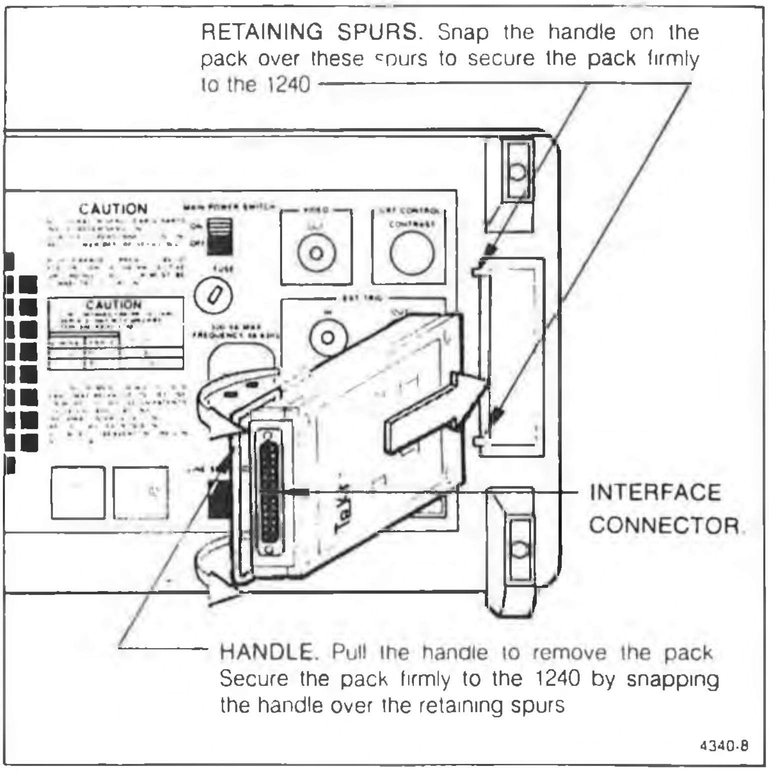


fig. 3-14. The COMM Pack installation

Fig. 3-15 shows a typical COMM port control menu display. The port status, GPIB address, and message termination type can be selected by using the COMM port control menu.

* GPIB port status: Valid selections are ONLINE and OFFLINE, in OFFLINE, no communication occurs between the controller and 1241, (before changing any other parameters, the 1241

must be OFFLINE). When the 1241 goes ONLINE, it sends a service request to the controller, notifying the controller

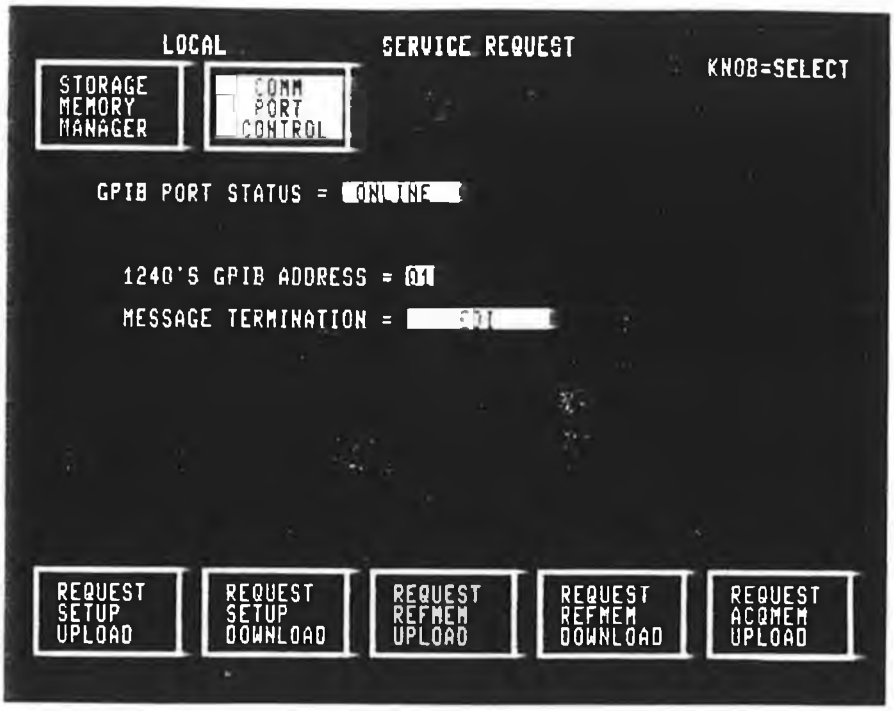


fig. 3-15. A Typical COMM Port Control Menu Display

of its ONLINE status.

* 1241's GPIB Address: Valid addresses are 0-30.

* Message Termination: Valid types are EOI, and LF OR EOI.

If EOI is selected, messages are terminated by sending EOI concurrent with the last byte of the message. During message reception, receiving an EOI is the only recognized message terminator. if LF OR EOI is selected, a CR followed by LF concurrent with EOI is sent as a message terminator.

In this project the address (1) was selected for the 1241 Logic Analyzer and the EOI as a message terminator.

CHAPTER 4

APPLICATION OF EXPERT SYSTEM IN DIAGNOSTICS

4.1. Introduction

Diagnosis has been considered a task well suited to expert systems solutions since early in the short history of expert systems. This is because diagnostic problems are not clearly defined and not well suited to algorithmic solutions. The emphasis with expert systems has not been on specific problem solving procedures but on knowledge about the problem domain and general procedures that reason with this knowledge. Diagnostic tasks can easily be represented in this format [35].

The main components of diagnosis can be summarized as follows:

Givens

(1) A case of malfunctioning, unusual 'symptoms';

(2) A standard set of diagnostic tests.

Goals

(1) To fit case into known 'disease, fault' classes;

(2) To find probable causes of symptoms;

(3) To recommend treatment methods.

Constraints

(1) The tests may be numerous and difficult to select;

(2) The tests may be costly (in time or money);

(3) The tests may be unreliable.

Operations

(1) Deduce possible causes of symptoms.

(2) gather data about symptoms and characteristics of the case.

(3) Classify possible causes into disease or fault hypotheses.

(4) Distinguish competing hypotheses.

(5) Take account of the interactions of several causes.

[52]

4.2. Introduction to Artificial Intelligence

Artificial intelligence gives computers extra computing capability, allowing them to exhibit more intelligent behavior. Intelligence, the ability of a human being to acquire knowledge and apply it, means the capability of thinking and reasoning. To a certain degree, artificial intelligence allows computers to accept knowledge from human input, then use that knowledge through reasoning processes to solve problems.

The main element of any artificial intelligence application is knowledge, an understanding of some subject area obtained through education or experience. The problem that the computer cannot have experiences or learn as the human mind can. So it can acquire knowledge given to it by human experts. This knowledge consists of facts, procedures and information that has been organized to make it understandable and applicable to problem solving or decision making. Most knowledge bases typically concentrate on some specific domain.

After building the knowledge base, artificial intelligence techniques will be used to give the computer reasoning capability. So the computer will be able to think, reason, and reach conclusions based on the facts contained in the knowledge base.

4.2.1. Application of Artificial Intelligence

The artificial intelligence software can be adopted to any problem requiring its special qualities (a problem with no algorithmic solution) because of the unbelievable flexibility of the artificial intelligence process.

Non-numerical problems and problems with uncertainty which are often not suitable for the algorithmic process, are both easily solved with artificial intelligence techniques.

"With algorithmic software, the problem is guaranteed to be solved; with AI, there can be partial solutions or even no solution.

As a result, AI often fits the disorganized, imperfect real world better than conventional software because it can deal with shades of gray." [68]

The most important applications to artificial intelligence are:

(1) Games

(2) General Problem Solving

(3) Expert Systems

(4) Natural Language Processing

(5) Computer Vision

(6) Robotics

(7) Education

4.3. Expert Systems

The major use of artificial intelligence nowadays is in expert systems. Expert systems are the artificial intelligence programs that act as intelligent advisors or consultants. By using the stored knowledge in a specific domain, a non-expert user can solve problems and make decisions in a subject area nearly as well as an expert.

An expert system permits the knowledge and experience of one or more experts to be captured and stored in a computer. This knowledge can then be used by anyone requiring it. The purpose of an expert system is not to replace the experts, but to make their knowledge and experience more widely available. So the expert system permits non-expert users to increase their productivity and solve problems when an expert is not available.

An expert system consists of three major components: a knowledge base, an inference engine, and a user interface.

The knowledge base contains all the facts and ideas of a specific domain. The inference engine analyzes the knowledge and extracts conclusions from it. The user interface implements communication with the user and permits new knowledge to be entered into the knowledge base.

To use an expert system, the user starts up the expert system software. The expert system then asks the user various questions to collect some initial information about the problem to be solved. The user can key in the information requested or select it from alternatives presented in menu form. Once the expert system has the input it needs, it starts searching for a solution and reaching a conclusion.

Usually users do not trust expert systems so they direct them to explain the process by which the conclusion was reached.

An expert system can be created to help users troubleshoot and repair a complex devices. The various troubles and symptoms can be given to an expert system which then identifies the problem and suggests the solution.

Expert systems also can be used to aid in diagnosing medical cases. Symptoms and test results can be given to the expert system which then searches its knowledge base in an attempt to match these input conditions with a particular disease. This results in a conclusion about the illness and some possible suggestions on how to treat it. Such an expert system doesn't replace doctors, but helps them confirm their own decisions.

Expert systems have been used to help locate oil and mineral deposits.

4.3.1. Widely Used Expert Systems

There are few number of expert systems available now in operation they represent just a small percentage of all computer software in use. Some of these expert systems played an important part in artificial intelligence development. So they make good models for studying the structure of an expert system.

MYCIN

MYCIN is probably the best known of all the expert systems. It was designed by Edward Shortliffe of Stanford University in the mid-1970s. It is a medical expert system that diagnoses bacterial infections [47].

DENDRAL

DENDRAL is another early expert system it is a product of researchers at Stanford University. DENDRAL was developed to identify the molecular structure of unknown compounds [54].

XCON and XSEL

XCON is an expert system that helps configure large computer systems. Developed jointly by Digital Equipment Corporation (DEC) and researchers at Carnegie Mellon University (CMU), XCON helps translate a customer's order for a DEC VAX 11/780 series computer system into complete final system specifications.

XSEL is another DEC-CMU expert system with knowledge of VAX 11\/780 computer systems. XSEL was designed to help DEC sales persons select components for a VAX system.

Prospector

Prospector is an expert system that helps geologists locate ore deposits.

EL

This expert system analyzes electronic circuits consist of transistors, diodes, and resistors. A schematic diagram of the circuit is entered into the computer and EL analyzes it.

SOPHIE

SOPHIE was designed to help students learn to troubleshoot electronic circuits. The system presents a simulated electronic circuit and a problem, then permits the student to troubleshoot it [36].

DELTA

DELTA was Developed by the General Electric Company to assist maintenance personnel in locating problems in diesel electric locomotive engines.

FOLIO

This is an expert system that helps stock brokers in handling investments for their clients.

VP-Expert

VP-Expert represents the first low-cost expert system program. For the first time, personal computer owners can develop powerful expert systems on their own with little or no knowledge of a special programming language.

VP-Expert can create expert systems for providing advice for legal consultation, financial planning, medical diagnoses, or electronic fault diagnosis [65], [66].

4.3.2. Advantages of Expert Systems

The previous programs point out some of the main advantages of expert systems. Like other conventional software, expert systems offer some advantages as following :

(1) Permit non_experts to do the work of experts.

(2) Improve productivity by improving efficiency.

(3) Save time in accomplishing a specific objective.

(4) Simplify some operations.

(5) Automate repetitive, or complex processes.

Expert systems offer some additional advantages over conventional software, such as:

(1) Permit new kinds of problems to be solved.

(2) Capture and store valuable knowledge that might be lost due to the resignation, retirement, or death of an expert.

(3) Make expert knowledge available to a wider users.

4.3.3. Disadvantages of Expert Systems

The disadvantages of expert systems are summarized as follow:

(1) Expert systems are expensive.

(2) It is very difficult to develop an expert system and hard to find good experts.

(3) Most expert systems still must be implemented on a big mainframe or minicomputer.

(4) Expert systems are not 100% reliable. So that the user should always provide the final judgment.

These disadvantages are significant but certainly not impossible to overcome.

4.3.4. Expert System's Type

There are two basic types of expert systems: the stand_alone system and the embedded type.

(1) Stand-Alone system

In this case the computer is totally dedicated to that program, while it is running it.

(2) Embedded system

This program is just a portion of another larger program. First, the expert system is built into an algorithmic program. This is illustrated in fig. 4-1A.

Second, an algorithmic program is built into an expert system. This is illustrated in fig. 4-1B.

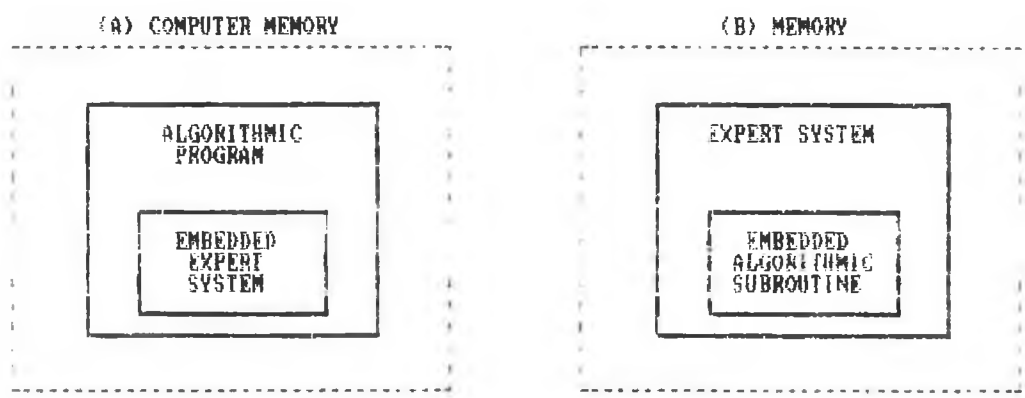


fig. 4-1. Types of Embedded Systems

VP-Expert is a real example of the embedded system. It links to an external software package such as a TURBO C. system. During the inferencing process, the VP-Expert system may reach out and access data in TURBO C. to get enough knowledge to reach a solution. While these programs are not really embedded, the effect is the same. They usually share memory with the expert system and have program links to it.

4.3.5. Expert System Architecture

An expert system is simply an arrangement of software components that permits an expert's knowledge to be used by others for problem solving and decision making in a specific domain.
The main components of an expert system are the knowledge base, the inference engine, the data base, and the user interface. A general block diagram is shown in fig. 4-2. Each of these elements will be explained in the following sections.

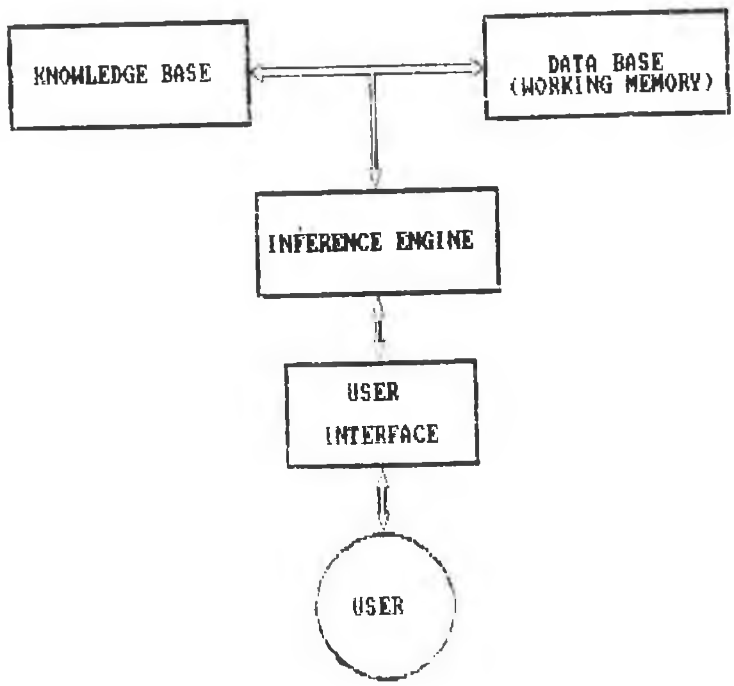


fig. 4-2. A General Block Diagram of an Expert System

4.3.5.1. The Knowledge base

The knowledge base is the heart of any expert system. There are different ways for representing knowledge in expert system, such as frames, semantic networks, and production rules. It is proved that production rules is the best way for representing knowledge.
"It has been determined through considerable experience that one of the best methods of knowledge representation for expert systems is

production rules. Most commercial and experimental expert systems use the popular IF_THEN rules format." [68]

Production Rules

Production rules are generally easy to write, and it is quick to built the desired knowledge base.

Rules is formatted into two parts. The first, the left-hand side of the rule, (IF) part, states some premise or condition. The second, the right-hand side of a production rule, (THEN) part, states a conclusion or action that will take place if the conditions on the left-hand side of the rule have been met. When the right-hand side of the rule is implemented, the rule is said to be fired. As an example of representing knowledge by using production rule:

IF the patient has headache

THEN the patient needs aspirin, CF .7

Each rule is made up of clauses. There is one IF clause and one THEN clause to every rule. The IF part of the rule may contain more than one clause. These are called compound clauses, and they are linked by AND or OR.

The certainty factor (CF) is a number between 0 and 1 that indicates the confidence in the validity of the conclusion.

While each production rule represents an individual piece of knowledge, it is usually related to many other rules. The rules link together to establish a line of reasoning. This collection of rules will form the knowledge base.

Finally, with production rules it is very easy to modify or add a new knowledge to the knowledge base.

4.3.5.2. The Data Base

The data base contains a broad range of information about the current status of the problem being solved. It is a portion of working memory where the current status of the problem-solving process is stored. Also the data base records facts about the problem. Initially, the Known facts are stored there. Then the new facts, which picked up from the inference process, are added. The initial conditions of the problem to be solved are also stored in the data base.

Usually, the expert system asks the user for some beginning input. This information gives the expert system a starting point to begin the search process.

The inference engine begins its search, matching the rules in the knowledge base against the information in the data base.

4.3.5.3. The Inference Engine

The inference engine is software that implements a search and pattern_matching operation. It examines the rules in a particular sequence looking for matches to the initial and current conditions given in the data base. As rules matching these conditions are found, the rules are fired.

As the rules continue to fire, they will reference one another and form an inference chain. Each time a new rule is examined, it is checked against the current status of the problem solution stored in the data base. The firing of a particular rule may add new facts to the data base. This gives the inference engine additional information to go on. This process continues until the solution is found.

The inference engine can take two basic approaches to search for an answer. These are forward and backward chaining.

Forward Chaining

In this case, the inference engine attempts to match a fact in the data base to the situation stated in the IF part of the rule. Once a fact has been matched, the rule is fired. The action stated could produce a new fact that is stored in the knowledge base. This new fact may then be used to search out the next appropriate rule. This searching and matching process continues until a final conclusion rule is fired.

Backward Chaining

In this case, the inference engine starts with the hypothesis in the data base. Then it begins examining the THEN parts of rules looking for a match. The inference engine searches for evidence to support the hypothesis originally stated. If a match is found, the data base is updated recording the conditions that the rule stated as necessary for supporting the matched conclusion. The chaining process continues with the system attempting to match the right side of the rule against the current system's status. The corresponding IF sides of the rules matched are used to generate new intermediate hypothesis which are recorded in the data base. The backward chaining continues until the hypothesis is proved.

4.3.5.4. User Interface

The user interface is a piece of software that lets the user communicate with the system. It asks questions or presents menu choices for entering initial information in the data base. It provides a menus of communicating the answer or solution once it has been found. Any intermediate communications during the problem_solving process are taken care of by the user interface. The clauses used in the rules are used as outputs with appropriate prefaces in simple systems. Some expert systems also, as VP-Expert, can include blocks of text with each rule and at the beginning or the end of the expert system. These are used to provide additional information or explanations.

4.3.6. Expert System Features

Expert systems are more useful if they have some additional features. These include an explanation facility, ease of modification, transportability, and adaptive learning ability.

4.3.6.1. Explanation Facility

Usually the first time users of the expert system are surprised at how quickly it comes up with a conclusion. They clearly don't believe it. Users frequently want to know just how the expert system arrived at that answer. Most of the expert systems have a means for explaining their conclusion. Typically, this takes the form of showing the rules involved in the decision and the sequence in which they were fired.

When users want to know the expert system's line of reasoning, they can read the rules and follow the logic themselves.

The explanation facility is important because it helps the user feel more comfortable with the outcome.

VP_Expert has very powerful explanation facility. These include: BECAUSE, HOW?, WHY?.

BECAUSE

The because keyword appears in the rule base. It provides an explanation of the expert system's rules during execution. It lets us add reasons to the rules in a rule base. These reasons describe how the rule works and why the rule requires specific answers from the user.

HOW?

The HOW? command lets us ask how VP-Expert assigned a specific value to a variable. If VP-Expert assigned the value to a variable, the HOW? command displays the BECAUSE text. If the rule does not contain a BECAUSE keyword and text, VP-Expert displays the rule.

WHY?

The WHY? command lets us ask VP-Expert why it is asking a specific question. If the rule contains the BECAUSE keyword and text, VP-Expert displays the BECAUSE text. If the rule does not contain the BECAUSE keyword, VP-Expert displays the rule. [65]

4.3.6.2. Ease Of Modification

In domains where rapid changes take place, it is important that some means be provided for quickly and easily incorporating this knowledge. If the expert system was developed using modern development tools, it is a simple matter to modify the knowledge base by writing new rules or removing rules.

4.3.6.3 Transportability

The wider the availability of an expert system the more useful it will be. The more different types of computers for which the expert system is available, the more widely the expertise can be used.

4.3.6.4 Adaptive Learning Ability

This is an advanced feature of some expert systems that allows them to learn from their own use. As the expert system is being operated, the inference engine will draw conclusions that can produce new knowledge. This new knowledge is stored temporarily in the data base, but in some systems they can lead to the creation of a new rule which can be stored in the knowledge base and used again in a future problem.

4.3.7. Uncertainty

Expert systems has the ability to deal with uncertain information. If an expert system, in collecting its initial inputs, ask a question for which the user does not has answer, the user can simply say that he does not know. Expert systems are designed to deal with inputs such as this.

There are several methods of dealing with uncertain information. In rule based expert systems, numerical factors indicating the probability of conclusion are used as a measure for uncertainty. These numerical factors are known as certainty factors (CF).

4.3.8. Fuzzy Logic

Fuzzy logic is another method of dealing with uncertain knowledge. It is a system conceived by computer scientist Lotfi Zadeh for dealing in unreliable information. In this method, an attempt is made to assign numerical ranges with a possibility value between zero and one to concepts such as tall, good, hot, and other elements with values that are hard to determine. [82]

4.3.9. Expert System Applications

The applications that fit the way an expert system represents knowledge will be explained, in details in this section, to determine whether a particular application is suitable for an expert system. Expert systems are definitely not suitable for all situations.

4.3.9.1 Control

The computer, in this application, is interfaced to other system. There are two basic types of control, open loop and closed loop. The computer, in an open loop system, follows step_by_step procedure to cause a particular type of behavior to occur in the system. A closed loop control uses feedback, that monitors the system status. This feedback is the key to use the expert systems in control applications, because it gives the expert system inputs to use in making decisions. With this kind of input information in addition to its own knowledge base, an expert system can adapt to changing conditions. It can also diagnose problems, and correct them by develop plans for overcoming them.

4.3.9.2 Debugging

Debugging is the process of troubleshooting, that finds problems in a system and comes up with a corrective solution.

4.3.9.3 Design

Design information for certain types of products can be stored in an expert system. A user can then call the expert system to design the product.

4.3.9.4 Diagnosis

Expert systems can be used to diagnose a malfunction in a device or system. From this aspect they are similar to debugging expert systems except that they do not give a solution to the problem. A diagnosis system observes the behavior of the device or system and makes note of improper performance.

4.3.9.5 Instruction

An expert system domain can be used in teaching students how to solve problems in the field of expertise. The student can work through different problems with the expert system. By using the explanation subsystem, the student can notice the sequence of rules being used to reach a conclusion. So the student can get an idea about what knowledge an expert needs and how it is used to solve problems.

4.3.9.6. Interpretation

Interpretation is one of the best applications for an expert system. Interpretation systems are given inputs that consist of observations and other data. Then, using its knowledge base and inferencing system, the interpretation system attempts to deduce a particular situation from the input data. It attempts to explain the situation which it represents.

4.3.9.7. Planning

A planner attempts to come up with a method or approach that will achieve a goal. Given the objective and starting point, as well as some possible intermediate steps, an expert system can produce an optimum plan for achieving that goal. For very complex projects, planning is a difficult human task. But for an expert system with the appropriate knowledge and inputs, planning is an easy task.

4.3.9.8. Prediction

Predicting means to foretell the future. A prediction system receives input data about a given situation. Then it deduce the outcomes. Weather forecasting is good example of an expert system of this type.

4.3.9.9. Repair

Repair is the process of returning a broken device to its original state. A repair expert system implements automatic repair of the system. That accomplish by building the capability of diagnosis, debugging, and planning into it. The system locates the trouble, suggests the fault candidate list, works out a plan, and then implements it. [68]

4.4. Electronic Fault Diagnosis

Diagnosis is concerned with producing a hypothesis to explain why the observed behavior of a system is different from its expected behaviour. This definition of diagnosis is particularly suitable in the area of electronic fault diagnosis.

In general, diagnosis has two tasks. The first task is to find a hypothesis (a fault) that explains the malfunctioning of the device or to produce a hypothesis (a disease) that explains a particular set of symptoms.

The second task in both cases is the extraction of extra information that is required to formulate the hypothesis.

There are three stages in the diagnostic process:

Abduction: arrive at a hypothesis to explain the symptoms.

Deduction: derive experimental outcomes of the hypothesis and carry out tests.

Induction: conclude the hypothesis to be true or false.

[77]

MYCIN system is one of the early, famous expert systems for medical diagnosis [47]. It is a backward chaining rule based system for diagnosing and treating infectious blood diseases. The knowledge about the problem domain is represented as **IF - THEN** rules, thus it is shallow knowledge.

This shallow knowledge representation is characteristic of diagnostic expert systems in medicine [46]. On the other hand, some of the expert systems developed for electronic fault diagnosis have been model based ([44] or [36]). This gives the fundamental difference in the nature of the reasoning involved in medical diagnosis and in electronic fault diagnosis. Medical diagnosis is based on models of the pathology of the system under diagnosis these are shallow models whereas electronic fault diagnosis is based on models of a correctly functioning circuit.

This distinction does not only exist in the way human experts perform these tasks but is also reflected in the way expert systems are designed to operate in these domains.

That mean that expert systems for medical diagnosis need to be based on shallow knowledge representations while those developed for troubleshooting electrical circuits can be based on either deep or shallow knowledge. An electronic fault diagnosis system can be model based or it can be built using shallow rules linking symptoms to causes [64].

There are two types of human experts, the design engineer and the test technician. Their reasoning strategies for diagnosis may be completely different. In addition that one expert might use different strategies at different stages. A technician will start off reasoning the first principles of what he knows about electronics and will proceed to learn shallow rules of thumb linking symptoms to possible causes as he becomes more familiar with the circuit. So he might operate like a qualitative model based system. A design engineer might be considered similar to a model based expert system. So he might operate like a quantitative model based system.

The architecture for a knowledge based system for electronic fault diagnosis should reflect one of these approaches [35].

4.5. Shallow Knowledge

Conclusions are drawn directly from facts that describe the problem in shallow reasoning. Shallow knowledge is best considered as 'rules of thumb' that describe a particular problem domain. These would be simple rules linking symptoms to causes in diagnosis. Human experts may acquire their knowledge at this level or they may acquire them at a deeper level where the knowledge is based on a model of the system and the reasoning is done from the first principles in this model.

If a particular problem is unsolvable with the shallow knowledge available in a system then the knowledge base can be extended to cover that problem. This extension will cover that particular problem only. This explain the main problem with shallow representations. A shallow knowledge based system will not perform well on cases not considered in its construction.

4.6. Shallow Knowledge Based Systems

Shallow knowledge based systems are usually rule based systems. The knowledge of a particular domain is represented as IF-THEN rules. The shallow knowledge is considered to have some advantages including:

(1) Shallow rules are an easy way of representing the domain knowledge of an expert.

(2) The knowledge base can be easily extended by adding new rules.

(3) The use of a backward chaining rule based system facilitates the development of a user interface as the inference engine will automatically ask the user for any information that is not available to the system [39].

(4) It is easy to have a rule based system explain its line of enquiry.

The advantages and disadvantages of shallow rules based systems can be considered using the PROCESSEX system which is an expert system for fault diagnosis in digital electronics. It is written in EMYCIN which is a backward chaining system.

PROCESSEX contains about one hundred and fifty rules which are written in both English and the Lisp-like code used by the system. EMYCIN use the English representation in its explanation facility. The rule set has a reasonably complex structure as shown in fig. 4-3. This complex structure inherent in the rule base is characteristic of most rule based systems and contradicts that rule base systems are easily extended. If an extra rule is to be added to the system then care must be taken to ensure that it fits into the existing structure. [64]

4.7. Deep Knowledge

A model based system is more general than the equivalent shallow knowledge based system as specific heuristics which handle

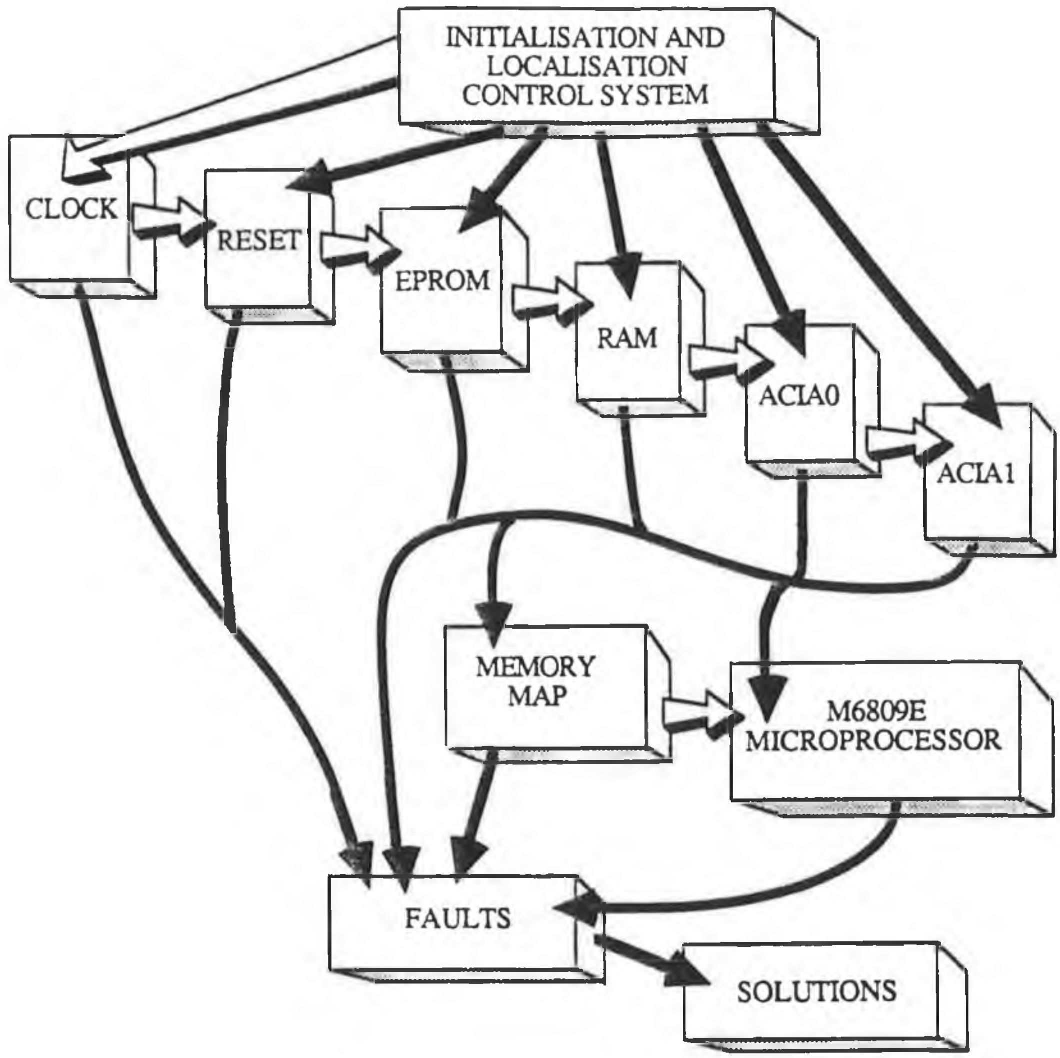


fig. 4-3. The Complex Structure of PROCESSEX

individual cases need not be encoded. It should be able to perform well on problems not clearly considered in its construction. At the same time a model based system should perform reasonably well on problems on the periphery of its knowledge.

The reasoning mechanism should be able to draw conclusions in all cases from the first principles described in the model. The control structures required are more complex because of this generality. They also take longer to execute. That because of this complexity, and because the inference chains are longer due to the small size of the knowledge [63].

4.8. Deep Knowledge (model) based systems

Knowledge elicitation difficulties with shallow knowledge based systems are probably a factor contributing to the amount of effort invested in producing model based expert systems for electronic fault diagnosis. This is an important difference between knowledge based systems for medical diagnosis and those for electronic fault diagnosis.

Most medical diagnosis systems are shallow knowledge based systems because deep models of the problem domain would be very complex and are not used by human experts. Instead experts use models of the various pathologies of the system. This is reflected in an expert system for diagnosis of glaucoma called CASNET. CASNET stands for causal-associational network and is semantic network based system [48].

The equivalent to these pathological models in electronic fault diagnosis are fault models; these are models of the various different types of fault that can occur in a circuit.

Classes of failure, as used by Davis, are general fault models but are peripheral to the model based diagnosis system [41]. Davis main model is a model of the structure and behaviour of the system.

The advantages and disadvantages of deep knowledge based systems can be considered using the DART system.

Consider the full-adder represented by the circuit diagram in fig. 4-4.

The structural information about this circuit is represented by Lisp-like expressions, of which the following are examples:

```
(XORG  X1)
(CONN  (OUT 1 X1) (IN 1 X2))
```

The first of these expressions indicates that X1 is an OR-gate. The

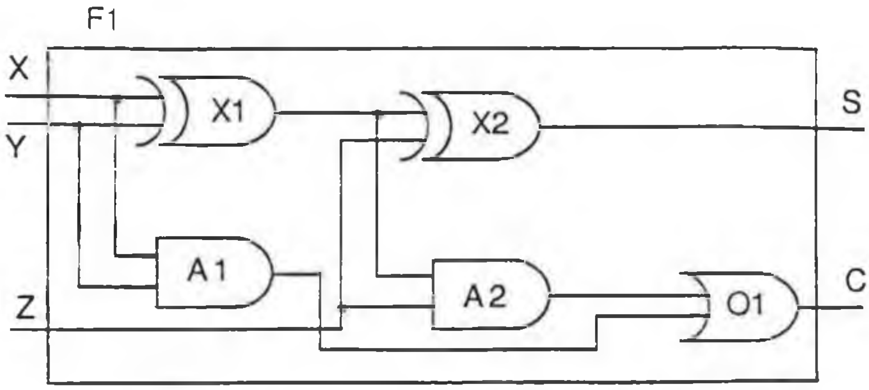


fig. 4-4. Schematic of the Full-Adder in DART

next one indicates that the output of X1 is connected to the first input of X2. The behavioural information is represented as follows:

```
(IF (AND (ORG d) (VAL (IN 2 d) t ON))
  (VAL (OUT 1 d) t ON))
```

If the device is an OR-gate and the second input is on then the output is on. The process for tracing through this model to find a fault is described in fig. 4-5.

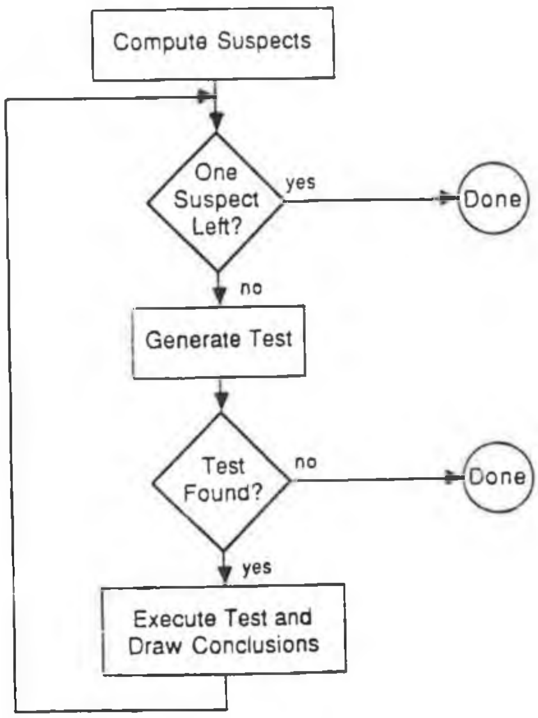


fig. 4-5. Flowchart of the DART Diagnostic System

The computation of suspects produces a suspect set which has the following form:

(OR (NOT p1) ....(NOT pn))

Where pi is a statement from the circuits design description not known to be true. For instance, if the diagnosis had reached a stage where it knows that either of the XOR-gates, X1 or X2 is broken, the set will be:

(OR (NOT (XORG X1))....(NOT (XORG X2)))

Clearly the diagnosis is complete when the set of suspects contains only one entry. The set of suspects is reduced by generating distinguishing tests between them.

The difference between qualitative and quantitative models is not important with digital electronics where the basic one-zero model is the only reasonable representation. On the other hand analog devices have an infinite number of possible states so there is a considerable difference between quantitative and qualitative models.

These digital fault diagnosis systems are presented working on simple circuits. Real digital circuits are more complex than this. So it seems likely that pure model based expert systems for fault diagnosis will be estimately very difficult for the expected future. This is very similar to the case with analog circuits where quantitative models are very complex. [24], [40], [42]

4.9. Advantages and Disadvantages of Shallow and Deep Knowledge Based Systems

The advantages and the disadvantages of both, deep and shallow knowledge based systems, will be presented in summary in this section.

4.9.1. Advantages and Disadvantages of Shallow Knowledge Based Systems

The advantages of the shallow knowledge based systems are:

(1) Speed of execution.

(2) Ease of representation of expert knowledge as shallow rules.

The disadvantages of the shallow knowledge based systems are:

(1) Difficult to extend the knowledge base.

(2) Will not perform well on problem cases not considered in system construction.

(3) Knowledge may be distributed throughout the system and therefore difficult to alter.

4.9.2. Advantages and Disadvantages of Deep Knowledge Based Systems

The advantages of the deep knowledge based systems are:

(1) Should be able to perform reasonably well on cases not considered explicitly in the construction of the system.

(2) Easily altered to operate in another problem situation by changing the model.

The disadvantages of the deep knowledge based systems are:

(1) Slow of execution.

(2) Reasoning mechanism will be difficult to generate because of its generality. [35]

4.10. Systems Incorporating both Deep and Shallow Knowledge

A deep knowledge based system would be the ideal solution were it not for the problem of performance. It seems unavoidable that a system that reasons about a circuit from first principles will be slow. The obvious solution is to incorporate some shallow knowledge into the system in order to 'short cut' some of this reasoning from first principles. This approach is supported in [38] and [45].

This project will be described as an example of a knowledge based system for electronic fault diagnosis incorporating both deep and shallow knowledge.

4.10.1. EXP-Test System

Electronic fault diagnoses is considered a suitable area for expert systems applications and the research in this area is well documented [38], [41], [45]. It is considered that the best approach to the problem of generating expert systems for troubleshooting electronic circuitry is to incorporate both deep and shallow knowledge in the system [45]. This approach recognizes the usefulness of both types of knowledge and utilizes both in the troubleshooting process.

Given symptoms of misbehaviour, the expert system must be able to determine the structural defects responsible for the fault from the deep knowledge base [55]. The structural knowledge has two components: a functional part and a physical part. The functional part is similar to the schematic of the circuit while the physical part corresponds to the circuit layout.

EXP-Test System was developed as a system for electronic fault diagnosis. It has the advantages of the both types of knowledge, shallow and deep knowledge, (The shallow knowledge is the knowledge about fault procedure and the deep knowledge is the knowledge about electronic circuitry).

The following example will describe the way EXP-Test System represent the shallow and deep knowledge (this example checks the output of the transformer):

```
RUNTIME;        ! This  statement  eliminates the Rules and Values
                ! windows when the user runs the rule base.
ACTIONS         ! This statement and FIND clause define the steps
                !  for solving a problem.
      DISPLAY "                      WELCOME TO
                                   EXP-TEST SYSTEM
                           SYSTEM FOR TESTING CNC MACHINE
                                      <PRESS ANY KEY TO START>~"
CLS             ! This clause clears the consultation window.
FIND THE_PROBLEM;
RULE    POWER_SUPPLY
IF      CHECK_TRANSFORMER = YES
THEN    CALL \TC2\SKETCH1
        CALL \TC2\SKETCH2
        CALL \TC2\EXP_T1
        LOADFACTS  FILE9
        FIND THE_TEST
        THE_PROBLEM = FOUND;
RULE    POWER_SUPPLY1
IF      delta1 >= (stander1)
THEN    WOPEN 1,15,1,8,60,4  ! This clause define the position,
                        ! size, and background color of a window.
        ACTIVE 1  ! This clause displays a window on the screen.
        DISPLAY
            "THERE IS A PROBLEM  IN  THE  TRANSFORMER, I SUGGEST
             YOU TO REPLACE IT.
             PRESS <ENTER>, THEN <Q> TO EXIT.~"
        WCLOSE 1 ! This clause removes a window from the screen.
        RESET delta1      ! The RESET keyword sets the value of a
```

```
            RESET stander1    ! variable to unknown.
            THE_TEST = FOUND
    ELSE    CLS
            DISPLAY "THE TRANSFORMER IS OK, <PRESS ANY KEY> TO
                     CONTINUE TESTING THIS DEVICE.~"
            RESET  delta1
            RESET  stander1;
                 .
                 .
ASK     CHECK_TRANSFORMER   : "TO CHECK THE TRANSFORMER:
                DISCONNECT THE TRANSFORMER'S OUTPUT FROM (J9).  THEN
                PUT THE OSCILLOSCOPE'S PROBE AT POINT (1).
                TO LOCATE POINT (1) AND CONTINUE THE TEST,
                CHOOSE YES, AND PRESS  <ENTER>. OTHERWISE CHOOSE NO,
                AND PRESS <ENTER>.";
CHOICES   CHECK_TRANSFORMER : YES, NO;
```

The previous example contain, the two types of knowledge deep and shallow knowledge.

The shallow knowledge in EXP-Test System was represented in nearly one hundred of the IF-THEN rules. These IF-THEN rules gave the system an easy way to represent the domain knowledge of the expert in the maintenance, also it gave the system a high speed execution and the ability to extend the system easily, by adding some rules.

The system backward-chains through these rules with the goal of discovering which of fault candidates is faulty. The clauses on the left-hand side of the rules must be found true in order to prove the clause on the right-hand side. If the information in the clauses on the left-hand side is not known to the system then it will query the user for that information. So the system get the extra information that it needs to isolate a fault by asking the user.

The ultimate problem, or "goal" of consultation is defined in a FIND clause in the ACTIONS block of the knowledge base. This clause instructs the inference engine to FIND a value for a given variable, which we call the "goal variable". The goal variable named in the FIND clause of EXP-Test System's ACTIONS block is THE_PROBLEM. Once THE_PROBLEM has been identified as the goal variable, the inference engine searches the knowledge base for the first rule that can assign a value to the goal variable. Since the goal variable is THE_PROBLEM, the inference engine looks for the first rule containing the variable THE_PROBLEM in its conclusion. This is the rule POWER_SUPPLY. Once the rule is found, the inference engine looks at the first variable named in the premise of the rule. If it does not know the value of the variable, it looks for the first rule containing that variable in its conclusion.

The only variable named in the premise of POWER_SUPPLY is CHECK_TRANSFORMER. Since the inference engine doesn't know the value of CHECK_TRANSFORMER, it scans the knowledge base for the first rule which might provide the value, in other words, the first rule which might contains the variable, THE_TRANSFORMER, in its conclusion. Because there is no such rule in the knowledge base, the inference engine (after scanning all the rules for one that can assign a value to the variable THE_TRANSFORMER) looks for an ASK statement that can provide a value for THE_TRANSFORMER. ASK statement will prompt the user for information not contained in the knowledge base. Then the inference engine will search for a CHOICES statement naming the same variable, which creates a menu of options to accompany the question during a consultation. In the studied case the ASK statement will ask the user:

TO CHECK THE TRANSFORMER:

DISCONNECT THE TRANSFORMER'S OUTPUT FROM (J9). THEN PUT THE OSCILLOSCOPE'S PROBE AT POINT (1). TO LOCATE POINT (1) AND CONTINUE THE TEST, CHOOSE YES, AND PRESS <ENTER>. OTHERWISE CHOOSE NO, AND PRESS <ENTER>.

The choices will be: YES NO

Here the user makes choice, so if he chooses YES. YES becomes the value of CHECK_TRANSFORMER. Since the inference engine has found a value for CHECK_TRANSFORMER, it will go back to rule POWER_SUPPLY, to give CHECK_TRANSFORMER the defined value.

Since the value of CHECK_TRANSFORMER is YES, the inference engine will fire POWER_SUPPLY. When POWER_SUPPLY is fired the inference engine will calls first of all the SKETCH1 file, which draw the schematic diagram of the circuit, then the inference engine calls the SKETCH2 file, which draw the layout diagram of the circuit, then the inference engine calls the EXP_T1 file, which get the signal from the peripheral instruments and compare it with the reference signal, and give the result value to variable delta1, and calculate the allowable tolerance and give its value to stander1, then EXP_T1 program saves delta1 and stander1 in a text file (FILE9). The inference engine then gate FILE9 from TURBO C., and give the value FOUND to the goal variable THE_PROBLEM, which just was an index to tel that the match is found. The FIND statement inside POWER_SUPPLY rule, lead the inference engine through the knowledge base, by giving a new goal.

After that the inference engine go to POWER_SUPPLY1 rule, which contain the goal variable in its conclusion. Depending on the comparison between the value of 'delta1' and the value of 'stander1' the inference engine will fire the POWER_SUPPLY1 rule or it will not.

The deep model should contain a structural and behavioural description of the circuit.

The deep knowledge was represented in the graphic accompany to each test applying on the circuit, this graphic has two dimension the physical dimension (layout diagram), and the functional dimension (schematic diagram).

In addition to representing the deep knowledge in the graphic, the deep knowledge can be represented in the intelligence of the technician and the production rules. As mentioned, with production rules, it is very easy to modify or add a new knowledge to the knowledge base. The key to the success of an expert system is the integrity of its knowledge base. If the knowledge base is incomplete, the system will be a poor problem solver and it may lead to wrong solutions. Therefore, it is important that the knowledge base be kept up to date at all times. In electronic fault diagnosis a new knowledge is being added continually. Experts continue to gain different experiences and improved problem_solving methods. All of this should be incorporated regularly into the expert system to keep it up to date. The rule format makes this easy because it breaks the knowledge down into small pieces. So an old rules can be rewritten to accommodate changes. Also a new knowledge is added by simply writing new rules and storing them in the knowledge base.

CHAPTER 5

INTEGRATING EXPERT SYSTEM AND ON-LINE TEST

5.1. Introduction

The term 'expert system' refers to a computer program that is largely a collection of heuristic rules (rules of thumb) and detailed domain facts that have proven useful in solving the special problems of some technical field. Expert systems to date are an outgrowth of artificial intelligence (AI), a field that has for many years been devoted to the study of problem-solving using heuristics, to the construction of symbolic representations of knowledge about the world, to the process of communicating in natural language, and to learn from experience. Expertise is often defined to be that body of knowledge that is acquired over many years of experience with a certain class of problem. One of the hallmarks of an expert system is that it is constructed from the interaction of two very different people: a domain expert, a practicing expert in some technical domain; and a knowledge engineer, an AI specialist skilled in analyzing an expert's problem-solving process and encoding them in a computer system. The best human expertise is the result of years, perhaps decades, of practical experience, and the best expert system is one that has profited from contact (via the knowledge engineer) with a human expert [52].

Repair of electronic systems and electromechanical equipment is a domain well suited to expert systems technology. Expert systems have been built in a wide variety of domains [78] including systems that specifically deal with repair. For example, expert systems for troubleshooting have been developed for computer installations and minicomputers. Dart [24], [54] is a system used to assist a technician in finding faults in a computer system. Two versions of Dart have been recently reported that use different expert system tools as basic building blocks. Emycin [79], a rule-based system, was employed in the first Dart version [24], while a meta-level reasoning system [80] was used in the second system [54]. IDT [56], an Intelligent Diagnostic Tool, is a system designed to assist in identifying faults in PDP 11/03 computers. This system was built using OPS-5 [81], a rule-based system developed at Carnegie-Mellon [53].

In this research on repair systems. It was hypothesized that expert systems can capture the repair knowledge of the best qualified human experts in a particular domain. A successfully implemented service advisory system would allow a technician with only a brief training period to conduct expertly guided service procedures. Reduction of service time and overall maintenance costs, improved and uniformly applied service strategies, and automated record keeping are among the expected benefits of the use of such an expert service advisory system.

5.2. Using Expert System in Building Intelligent Workstation

An expert system is the intelligent element among the main elements which assist in building an intelligent workstation. Using an expert system in constructing an intelligent workstation will mean implementing the artificial intelligence techniques on this workstation. This will give the workstation an intelligence attribute and will lead to a powerful workstation.

Electromechanical systems such as CNC machines have service manuals which describe repair procedures at a certain practical level of detail, eg., about board or electromechanical unit replacement. Repair of such systems is normally accomplished by direct replacement of Field Replaceable Units (FRUs). Detection and replacement of defective FRUs are the basic objectives of the repair task in these systems. Depending on the type of system serviced, service manuals vary considerably in complexity. For some systems, complex decision trees and flowcharts are part of the manual while in other systems only rudimentary information on repair is provided [53].

An expert's 'intuition' and efficiency are not found in service manuals and such information is normally not provided to the end_user. In fact, complex repair procedures are difficult to provide in a manual. For example, basic operations such as setting up hypothesis based on symptoms, proving hypothesis, and suggesting and verifying corrective actions are often complicated by the presence of subclasses or subproblems which require a changing focus of attention and/or changing strategies when solving a problem. Effective focusing mechanisms and flexible solution strategies cannot be readily suggested in a service manual due to lack of contextual information. However, these methods that a human expert readily performs can be implemented using artificial intelligence techniques [53].

That mean that the main task of the expert system is to capture the repair knowledge of the best qualified human experts in its knowledge base.

So the existence of the expert system inside the intelligent workstation gives a successfully implemented service advisory system which is the best solution to stop calling in the expert person every time a breakdown happened to any of the CNC machines in the workshop.

5.3. Improving the Productivity

By building the intelligent workstation which includes all the repair knowledge in its expert system's knowledge base, we can depend on a technician with a limited experience to solve any machine's breakdown problems. That will reduce the service time and overall maintenance costs, also that technician will increase his knowledge day by day by fixing machines using the intelligent workstation. If he adds the acquired new knowledge to the knowledge base of the expert system, he will be able to help himself and other technicians after him to achieve a quick fault diagnosis in the future. That of course will reduce the breakdown time and increase the productivity and the benefit to the whole plant.

5.4. Developing an Intelligent Mobile Workstation for On-Line Test.

As illustrated in fig. 1-1 the intelligent mobile workstation is a mobile unit, which can be easily moved from the service station to any CNC machine inside the workshop.

The total system configuration, which is shown in fig. 5-1, consist of:

(1) EXP-Test System.

(2) Graphics.

(3) Technician.

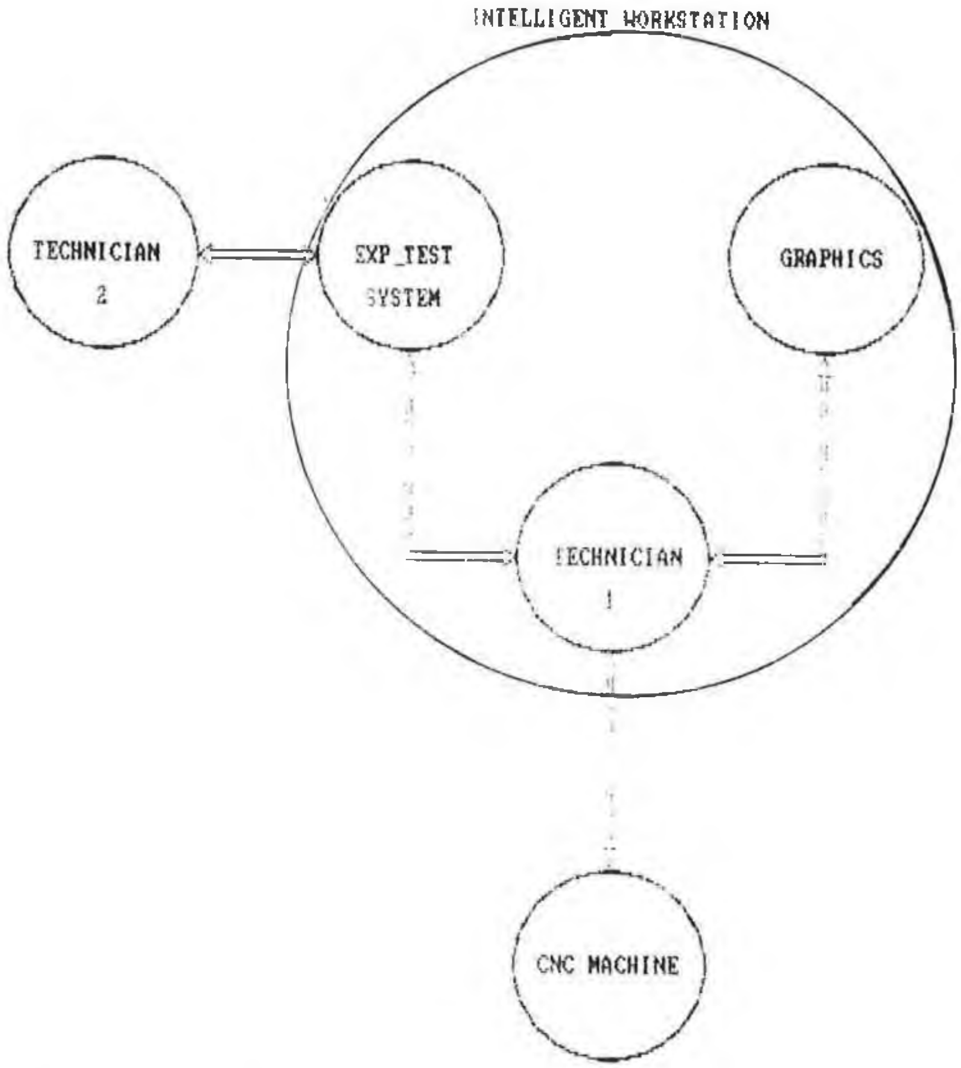


fig. 5-1. System Configuration of the Intelligent Workstation

Exp_Test System

The highly specialized piece of software that attempts to duplicate the function of an expert in some field of expertise. The program acts as an intelligent consultant or advisor in the domain of interest, capturing the knowledge of one or more experts in its knowledge base. Non_experts can then tap the EXP-Test System to solve repair problems, and make decisions in that repair domain.

Graphics

As mentioned the Graphic element in the intelligent workstation captures the important part of the deep knowledge of the system, (because the Graphic element is very important, we will devote a complete chapter to explain the integrating graphic CAD knowledge with expert system and on-line test).

Technician

The technician being considered is a practitioner with limited experience. This means a technician with only a brief training period on guided service procedures is suitable. So an expert with a high level of performance is not required.

The union of these three elements comprises the intelligent mobile workstation.

As mentioned, the intelligent mobile unit contains the shallow knowledge and the deep knowledge. It is very difficult to incorporate all the knowledge inside the knowledge base of the system in a short-time or during the creation of the system. So just the shallow knowledge is included in the production rules (IF-THEN) of EXP-Test System at the early stage. By using the system it would be essential for the user to refine the shallow knowledge and to add new cases which are encountered during the use of this system. The shallow knowledge is updated and it expands continuously while the system is being used. EXP-Test System contains nearly one hundred production rules, these rules contain the knowledge of the expert in the CNC machine. A two dimensional Graphic facility has been incorporated in this work, the physical dimension (layout diagram), and the functional dimension (schematic diagram) of the circuit board which is going to be checked.

The deep knowledge resides in the Graphics facility, which contain the structural information about the U.U.T. (unit under test) circuitry, and the intelligence of the technician. A very important part is played by this technician in analyzing the schematic diagram of the circuit and in defining the solution if a new case is encountered.

By integrating the EXP-Test System with the accompanying Graphics facility and with the intelligence and the knowledge of the technician, the intelligent mobile workstation will incorporate both deep and shallow knowledge. This incorporation will give the intelligent mobile workstation the capability of solving all the problems which might occur.

Now, let us ask this question:

How can that accomplish ?

In the case of a breakdown to a CNC machine, the user of the machine will call the service technician, who of course, immediately will move the intelligent mobile workstation to the location of the broken-down machine. There he will switch on the workstation and call EXP-Test System. EXP-Test System will begin to guide the technician to diagnosis the problem.

The user interface will help the EXP-Test System to ask the technician for some information and will ask him also to move the (Digital Oscilloscope's probe or the Logic Analyzer's probe), from one point to another, to get some test signals.

The Graphic facility has been integrated with EXP-Test System in such a way that EXP-Test System plots the desired circuit board on the screen. Then it shows the point which needs to be checked by EXP-Test System.

At this stage there are two possibilities:

1. If the type of fault, which is on the machine, is included within the knowledge base of EXP-Test System, EXP-Test System will define the problem and will print out the fault candidate list. This will depend on the information which is provided by the technician and on the result of comparing the test signals with the reference signals of each point.

The following example, explains how EXP-Test System asks the user for information, ask him also to move the oscilloscope's probe from point to another, and how it defines the fault candidate list:

When the user start EXP-Test System, it will display:

WELCOME TO

EXP-TEST SYSTEM

SYSTEM FOR TESTING CNC MACHINE

<PRESS ANY KEY TO CONTINUE>

The user must follow all the instructions which appear on the screen, so if he is ready for testing the machine, he will press any key on the keyboard. If that happened, EXP-Test System will display:

CHECK THE MAIN POWER PLUG (220) ACV,

IF THE MAIN POWER IS OFF, (MOVE THE CURSOR TO OFF AND PRESS <ENTER>, IF ON PRESS <ENTER>.

ON OFF

In this case the user should check the main power and notify

EXP-Test System, what is the result. If the main power is OFF, EXP-Test System will display:

SWITCH ON THE MAIN POWER AND BE SURE OF THE CONNECTION BETWEEN THE MAIN POWER AND THE DEVICE.

PRESS <ENTER> TO RETRY THE PREVIOUS TEST.

But if the main power is on, EXP-Test System display:

TO CHECK THE TRANSFORMER:

DISCONNECT THE TRANSFORMER'S OUTPUT FROM (J9).

THEN PUT THE OSCILLOSCOPE'S PROBE AT POINT (1).

TO LOCATE POINT (1) AND CONTINUE THE TEST, CHOOSE YES, AND PRESS <ENTER>. OTHERWISE CHOOSE NO, AND PRESS <ENTER>.

YES NO

If the user presses YES, EXP-Test System will do the following procedures. First of all, it will call SKETCH1 program from the TURBO C. system, TO draw the schematic diagram of the circuit. The user should press any key, then EXP-Test System will call SKETCH2 program from TURBO C. system. Also, this program draws the layout diagram of the circuit, and will cause the transformer area , and point (1) to flash, hence locating point (1). After that the SKETCH2 program will zoom onto the flashing (the transformer) area around point (1), and will display:

PRESS ANY KEY TO CONTINUE.

The user should put the probe at point (1), then he should press any key on the keyboard.

EXP-Test System will call EXP_T1 program, which as mentioned, will return two variables: delta1 and stander1.

If delta1 >= stander1 (if the received signal is wrong), EXP-Test System will tel the user that there is a problem in the transformer, so it will display:

THERE IS A PROBLEM IN THE TRANSFORMER, I SUGGEST YOU TO REPLACE IT.

PRESS <ENTER>, THEN <Q>, TO EXIT.

But if delta1 < stander1 (if the received signal is correct), EXP-Test System will display:

THE TRANSFORMER IS OK, <PRESS ANY KEY> TO CONTINUE TESTING THIS DEVICE.

At this stage if the user presses any key, EXP-Test System will display:

PUT THE OSCILLOSCOPE'S PROBE AT POINT (2),

IF YOU ARE READY, CHOOSE YES, AND PRESS <ENTER>.

OTHERWISE CHOOSE NO, AND PRESS <ENTER>.

YES NO

The Graphic facility has been connected just to point (1) (to prove the capability of integrating EXP-Test System with Graphic facility).

If the user chooses YES, after he puts the oscilloscope probe at point (2), EXP-Test System will call EXP_T2, which will get the signal at point (2) and return delta1 and stander1 to EXP-Test System.

If delta1 >= stander1 (if the received signal is wrong), EXP-Test System will display (inside flashing window):

THERE IS A PROBLEM IN THE (5V) OUTPUT, TO DEFINE WHICH COMPONENTS CAUSE THIS PROBLEM, YOU SHOULD MAKE TEST TO SOME OTHER POINTS BETWEEN POINT (1), AND POINT (2), <PRESS ANY KEY> TO CONTINUE.

But if delta1 < stander1, then EXP-Test System will display:

THE (5V) OUTPUT IS OK, <PRESS ANY KEY>, TO CONTINUE TESTING THIS DEVICE.

Assume that (delta1 >= stander1) in the previous case, (wrong signal is received at point (2)).

If the user presses any key, EXP-Test System will display:

PUT THE OSCILLOSCOPE'S PROBE AT POINT (3).

IF YOU ARE READY, CHOOSE YES, AND PRESS <ENTER>.

OTHERWISE CHOOSE NO, AND PRESS <ENTER>.

YES NO

If the user chooses YES, then EXP-Test System will call EXP_T3, which will get the signal at point (3), compare it with the reference signal and return delta1 and stander1. (If the user chooses NO, the system will wait until the user is ready and it will prompt him again).

If delta1 >= stander1, EXP-Test System will display:

THERE IS A PROBLEM BETWEEN POINT (3), AND POINT (1), TO BE MORE SPECIFIC, <PRESS ANY KEY> TO CONTINUE TESTING (5V) LINE.

But if delta1 < stander1, EXP-Test System will display the fault candidate list:

THE EXPECTED DAMAGED COMPONENTS ARE:

(1) U21.

(2) VR1.

(3) C30.

(4) C28.

(5) C5_7.

That work will take by experience, about 5 minutes.

2. If a new defect is encountered, EXP-Test System will be terminated after several trials and a message will be given showing that a new problem is being processed.

   The following example, explains how EXP-Test System ask the user for more information, and tell him that a new case is encountered:

   This example will check the drive motors of the U.U.T., to define if the malfunction is from one of the motors or from the motor's driver.

SWITCH THE PLOTTER (U.U.T.) OFF, PUT A BLANK PAPER, ONE PEN AT LEAST, (NO1). THEN PRESS (P1) & (P2), AT THE SAME TIME SWITCH ON THE PLOTTER. THE TEST PROGRAM WILL RUN, TESTING THE OPERATION OF EVERY MOTOR, THAT CALLED (THE INTERNAL TEST PROGRAM). BY LOOKING CAREFULLY TO THE MOTORS, PRESS <ENTER>, IF ALL OF THEM WORK PERFECTLY, OR MOVE THE CURSOR TO PROBLEM AND PRESS <ENTER>.

NO_PROBLEM PROBLEM

If the user finds a problem in the performance of any motor, he will select PROBLEM.

In this case EXP-Test System will print out on the screen:

ONE OF THE DRIVE MOTORS DOESN'T WORK, <PRESS ANY KEY> TO DEFINE WHICH ONE.

Now the system will ask the user:

MOVE THE CURSOR TO THE SUITABLE DRIVE MOTOR WHICH YOU NOTICED DOESN'T WORK AND PRESS <ENTER>.

PAPER_DRIVE_MOTOR PEN_DRIVE_MOTOR

PEN_SOLENOID CAROUSEL_STEPPER_MOTOR

If the user selects PAPER_DRIVE_MOTOR, EXP-Test System will display:

<PRESS ANY KEY>, TO DEFINE THE DAMAGED COMPONENTS.

When the user presses any key, the system displays:

PUT THE OSCILLOSCOPE'S PROBE AT POINT (ENCAX), OR (ENCBX). THEN ROTATE THE MOTOR MANUALLY, (BY USING ONE OF THE BUTTONS ON THE FRONT PANEL OF THE PLOTTER). IF THERE IS A SQUARE SIGNAL PRESS <ENTER>, IF THERE IS NO SIGNAL:

MOVE THE CURSOR TO NO, AND PRESS <ENTER>.

YES NO

If the user observes the square signal on the oscilloscope's screen, he will select YES. In this case EXP-Test System will display:

THE ENCODER IS OK, <PRESS ANY KEY>, TO CONTINUE TESTING THIS UNIT.

The user should press any key to continue, then the system displays:

SWITCH ON THE LOGIC ANALYZER, THEN CHOOSE YES, AND PRESS <ENTER>. OTHERWISE CHOOSE NO, AND PRESS <ENTER>.

YES NO

Now the system will call EXP_SET1 program, which will setup the Logic Analyzer. When the Logic Analyzer is ready, EXP-Test System displays:

CONNECT POD (0), CHANNEL (0), TO POINT (A), AND PRESS (START) BUTTON, ON THE LOGIC ANALYZER FRONT PANEL. WAIT TILL THE ANALYZER GET DATA ON IT'S SCREEN, THEN CHOOSE YES, AND PRESS <ENTER>. OTHERWISE CHOOSE NO, AND PRESS <ENTER>.

YES NO

At this point the expert system will call EXP_T11 program, which will get the signal from the Analyzer and compare it with the reference signal. Then EXP_T11 calculates the total difference, save it in variable delta1, and return delta1 to EXP-Test System.

If delta1 > 0 (wrong signal is received), EXP-Test System will display:

THERE IS A PROBLEM IN THE INPUT OF THE PAPER MOTOR DRIVER. THIS IS A NEW CASE, TO DEFINE WHICH COMPONENTS CAUSE THE PROBLEM, YOU SHOULD DO SOME OTHER TESTS TO THE INPUT AND OUTPUT OF (U6) GATE ARRAY. PRESS <ENTER>, THEN <Q> TO EXIT.

It is a very difficult job for an expert to know of all the types of problems which might occur to a machine. That means there is always a lack in the knowledge base of EXP-Test System. This lack can be compensated for by integrating EXP-Test System's knowledge base with the knowledge which is included in the Graphic subsystem and the knowledge of the technician.

So if EXP-Test System didn't come to any solution, the technician has to analyze the schematic diagram of the circuit, and try to find out the fault.

However, there are two reasons why this will not take too long. The first one, because EXP-Test System has already completed part of the analyses. The second, is the great help which will the graphics facility give.

When this technician comes up with the solution, he will update EXP-Test System which may be easily accomplished by adding the new case to it's knowledge base.

As mentioned, it is very easy to update EXP-Test System, by adding or modifying it, because it consists of many production rules (IF-THEN), ie. it is modular.

Up-dating EXP-Test System will provide new insights, different experiences, improve problem-solving methods, and increase the depth of the knowledge inside it's knowledge base at the same time. That will ease the task for the new technician in the case of absence, retirement or death of the previous technician.

The new technician (technician2 in fig. 5-1) will get the advantages of all the knowledge of the expert and the knowledge of the previous technician. He will find them inside the knowledge base of EXP-Test System.

CHAPTER 6

INTEGRATING GRAPHIC CAD KNOWLEDGE WITH EXPERT SYSTEM AND ON-LINE TEST

6.1 INTRODUCTION

Computers have been used to facilitate the design process since the early 1960s. Applications have included analysis of designs, simulation, and even the complete automation of some of the design processes. In the early 1970s, interactive computer graphics became a practical tool for supporting graphics-based design. Presently, the great majority of computer-based work-stations supporting design are of the interactive computer graphics type.

During the preceding twenty-year period the acronym CAD has been used to apply to any or all of the various application areas [72]. Computer aided design (CAD), or designing with the aid of a computer, has evolved to the point where each one will be exposed to its capabilities. Today, designing with the aid of a computer is not reserved for a chosen few. Anybody who has a popular personal computer could purchase a CAD package for a few hundred dollars and begin designing today [73].

6.2 CAD System

6.2.1 Overview

The objective of CAD is to increase productivity by utilizing computers in the design process. The meaning of productivity here is the ratio of labor hours required for a manual design function to the labor hours required if a computer is used to support the function.

CAD is a broad subject that fits into a broad spectrum of automated methods. The following diagram (fig. 6-1) shows the place of CAD in the spectrum of automated methods:

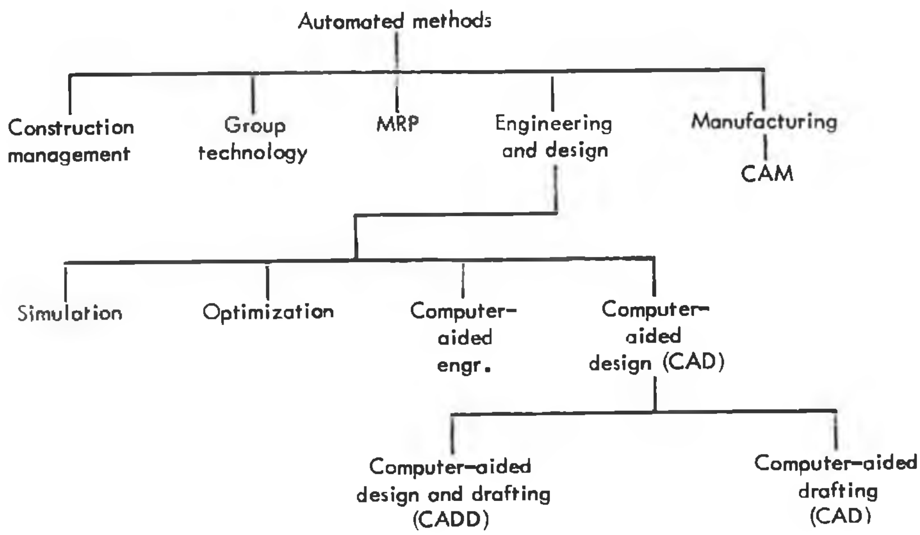


fig. 6-1. Place of CAD in the Spectrum of Automated Methods

CAD systems may be either two-dimensional or three dimensional, and they may be designed to provide either special-purpose or general-purpose applications. Although there is no strict use of the acronyms, CADD generally refers to "3D" (three-dimensional) systems that build a model of a design and tend to automate drafting. Computer-aided design (CAD), generally refers to "2D" (two-dimensional) systems that support drafting more directly.

A "special-purpose" application refers to a system dedicated to a specific narrow application, such as PCB boards, piping, and the like. Special-purpose CAD systems are often referred to as vertical systems.

A "general-purpose" application is a more general application such as a higher-level language that can be applied in a number of application areas. General-purpose CAD systems are often referred to as horizontal systems.

The great majority of CAD systems in use today are general-purpose 2D systems that are being applied to help drafting in any discipline: architecture, mechanical, civil, electrical, or electronic.

All CAD systems that in any way aid or automate drafting have the following components:

(1) Computer

(2) Graphics screen

(3) Graphics input such as a digitizer, keyboard, mouse

(4) Graphics output such as a plotter

Computer graphics is used in art, engineering, construction, manufacturing, business management, education, and many other areas. An architect can lay out or design a building using a computer and specialized graphic software. Animated cartoons and movies also take advantage of computers and graphic application program. One of the most basic forms of computer graphics is the bar chart or graph.

Simulation and training of pilots is another area where computer graphics has had great payoffs. With three dimensions representations of an aircraft's controls and an airport's landing strip, student pilots can learn the basics of flying without wasting expensive fuel. The simulation of an aircraft and an airstrip allow a student to experience the visual aspect of flying without having to step into a plane.

CAD Applications

A variety of CAD applications are currently in use on computers, from mainframes to personal computers. The industry is growing so fast that it's now necessary to specialize in a specific application area just to maintain a good knowledge of that field. One of The most important applications is:

Printing Circuit Design

If there is a single field in CAD that has benefitted most, it is the printed circuit design process. With the electronics industry becoming more competitive in developing new electronic products, the need for taking a designer's first sketches to final artwork is essential. A printed circuit CAD system can reduce everything from design to manufacturing the printed circuit boards in both time and cost. The productivity gained by using a printed circuit CAD system, compared to the manual method, can be up to four to one. A printed circuit CAD system automates repetitive and time-consuming tasks such as generating bills of materials, and manual design checking. The application programs for printed circuit design aid the designer in drawing the schematic diagram directly on the screen. A library of circuit symbols (available on-line) are used to place components on the schematic diagram. When finished, the drawing can be reproduced by a high-quality pen plotter or a simple printer.

A CAD method allows a design engineer to create on-line finished schematic diagram drawings. CAD systems contain pre-stored chip descriptions and component symbols in a library. These library symbols become the building blocks for schematic diagram design.

A CAD system must consist of a cpu, a storage device like a floppy disk drive, a monitor, an input device such as digitizer or mouse, and a keyboard. The software required is an operating system (for example, for a personal computer, MS-DOS); graphics software (for example: OrCAD/SDT III System); and a database to control the storage or the drawings produced. All the graphic installations have two basic parts: hardware and software. The hardware is the computer and its associated peripherals.

6.2.2. OrCAD/SDT III System

OrCAD/SDT III System is a complete and flexible schematic capture package. Easy to use menu driven commands help the user to create, edit, save, print, and plot electronic schematics. It is developed specifically to run on IBM personal computers and compatibles, OrCAD/SDT III supports most of the popular graphics boards, printers, and plotters. This eliminates the need for special, proprietary hardware by enabling the user to use standard output equipment.

The OrCAD/SDT III software package consists of the schematic drafting program DRAFT, a graphical library object editor called LIBEDIT, netlist, design check, part listing, and other utility and library programs.

DRAFT

DRAFT is the schematic drafting program that enables the user to create, edit and save schematic worksheet. The major features of DRAFT include:

(1) User definable template dimensions at 1 mil resolution.

(2) User definable text size.

(3) Eight part fields.

(4) Over 3500 Unique library parts.

(5) DeMorgan Equivalent parts.

(6) Placement of wires, buses, connectors, labels, and junctions.

(7) Real-time rubberbanding of wires and buses when objects are moved.

(8) Part rotation and mirroring.

(9) Moving, replicating, and deleting objects or blocks of objects.

(10) Powerful step-and-repeat command.

(11) Visible grid dots and angled bus entries.

(12) Automatic panning of the worksheet.

(13) Five zoom levels.

(14) Over 100 user-assignable macros.

(15) Unlimited levels of hierarchy.

(16) On-Line part browsing and library directory.

(17) PSpice analog simulation shell (PSpice not included).

(18) String searching.

(19) Vertical text placement.

(20) Suspension of session for DOS command execution.

(21) Supports "A" through "E", and custom size worksheets.

Part Libraries

Included with OrCAD/SDT III are extensive part libraries of the most commonly used devices in the industry.

Creating Custom Libraries

OrCAD/SDT III enables the user to create his own "custom" libraries, or modify existing ones, in two easy ways.
First, the user can invoke the graphical object editor called LIBEDIT. With this editor, the user can use commands similar to those of OrCAD/SDT III to construct or modify a part on the screen

and add it to a new or existing library.

Second, the user can use a text editor to create a library source file. A library source file is an ASCII text file that contains instructions in the OrCAD Symbol Description Language.

Utility Programs

OrCAD/SDT III's flexibility continues after the schematic design process with easy-to-use utility programs, including:

* ANNOTATE: This program scans a hierarchy or flat file and automatically updates all part reference designators.
* BACKANNO: The BACKANNO utility updates part reference designators in the user design. The input to the program, a list of old and new reference designators, is used to update the user schematic worksheets.
* CLEANUP: This utility checks the worksheet for wires, buses, junctions, labels, module ports, and other objects that are placed on top of each other.
* COMPOSER: If the user choose to create library parts using a text editor, COMPOSER is the library utility that converts the user custom library source files into the highly compressed library object files used by DRAFT.
* CROSSREF: This utility scans through the schematic files, gathers information for all parts used in the schematic files, and creates a cross reference listing that tells the user where each part is located.
* DECOMP: If the user choose to create library parts using a

text editor, DECOMP is a library de-compiler that enables the user to convert the OrCAD-supplied library object files to library source files.

* ERC: This is a utility that performs an electrical rules check of the user schematic worksheets.
* LIBARCH: This utility takes all the library parts used in the schematic files and makes a single library source, an archived library, containing only parts which are usable for those schematic files.
* LIBEDIT: This utility enables the user to create library components on the screen.
* NETLIST: This program generates a netlist of the worksheet signal and part connections.
* PARTLIST: This utility summarizes all the parts used in a schematic or group of schematic sheets.
* PLOTALL: PLOTALL plots a schematic or group of schematic sheets, in batch mode.
* PRINTALL: PRINTALL prints a schematic or group of schematic sheets, in batch mode.
* TREELIST: A program that scans a hierarchical organization of sheets to display the structure, sheet names, and sheet path names of the hierarchy.

6.2.3. OrCAD Plot File

During the last twenty years CAD (Computer Aided Design) has been used as a powerful tool in all engineering disciplines, specially in electronic engineering field, and printed circuit board design.

It was very important to incorporate CAD system in this work to achieve the interactivity, integrity and generality.

The OrCAD/SDT III, which is a computer aided drafting package, capable of drawing 2D diagrams, has been adopted.

OrCAD/SDT III package is a complete and flexible schematic capture package being used in the department. It has an easy to use menu driven commands, which help to draft and plot electronic schematics.

In order to integrate OrCAD with the EXP-Test System, it was necessary to adopt a neutral file, which can be produced by OrCAD/SDT III and read by an external program.

The plot files are the ideal files to be used in this process, because of its capabilities to represent all the entities, which can be drawn on the screen.

Also, these files are written in text files, which can easily be read and translated to other formats.

OrCAD/SDT III has the capability to create several types of OrCAD plot files. These files command plotters to create hardcopy of the drawings.

HP-formatted plot file has been chosen to be used as a neutral file in this integration process.

The main drawing entities represented in this format are as follow:

1. The line command: PA(x1,y1);PD;PA(x2,y2)

   this command, ask HP-Plotter to draw line between point(x1,y1) and point(x2,y2).

2. The text command: PA(x,y);LB *text*

   this command, ask HP-Plotter to write the defined text at point(x,y).

3. The circle command: PA(x,y);CI *r*

   this command, ask HP-Plotter to draw a circle with the radius r, at point(x,y).

4. The arc command: PA(x1,y1);AA(x,y),+/- start angle, the arc index.

   or:

   AA(x1,y1),+/- start angle,the arc index;AA(x,y),+/- start angle,the arc index.

   this command, ask HP-Plotter to draw an arc, at point(x,y). In the first format Point(x1,y1) define the start point (where the plotter's pen start drawing the arc). In the second format, which is encountered in the case of drawing a chain of arcs (coil), point(x1,y1) define the centre of the previous arc.

   +/- define the direction of the draw (+ for clockwise/- for unti-clockwise).

A software has been developed using TURBO C. Compiler to read the graphical format of the plot file to be used in EXP-Test System. This software is divided into two main parts. The first part of the program searches and extracts all the data of the drawing entities. After manipulating and modifying these data, it saves them in a data file to be used later.

The function of the other part of the software is to read this data from the data file, and prepare it so that it can be drawn on the screen. The screen is set to the graphic mode using a suitable scale according to the drawing. TURBO C. built-in functions have been used to create the schematic diagram on the screen. These functions are:

1. The line function: line(x1,y1,x2,y2)

   which draw line on the screen between point(x1,y1) and point(x2,y2).

2. The text function: outtextxy(x,y,*text*)

   which write the defined text on the screen at point(x,y).

3. The circle function: circle(x,y,*r*)

   which draw circle on the screen at point(x,y), with radius *r*.

4. The arc function: arc(x,y,start angle,end angle,*r*)

   which draw arc on the screen at point(x,y), beginning at the specified start angle, and finishing at the specified end angle, with radius *r*.

Several tests have been carried out using these programs, and as a result both, the drawings created by OrCAD/SDT III and the one extracted from the plot file were identical.

6.3. Programs

The software which reads the graphical format of the plot file and draws the graphics on the screen is divided into two main parts.

The first part is called the Search Program. The second part is called the Sketch Program.

Fig. 6-2 shows the flow chart of the Search Program.

The work of the Search Program is to search and extract all the data of the drawing entities, inside the plot file. Then the Search Program will save these data in a data file, after manipulating and modifying them.

In the beginning the Search Program opens the OrCAD plot file to read it, and opens a data file to save the modified data in it. Then the Search Program makes checks on the plot file to ensure that it is not empty. At this stage if the plot file is empty the search program will print on the screen:

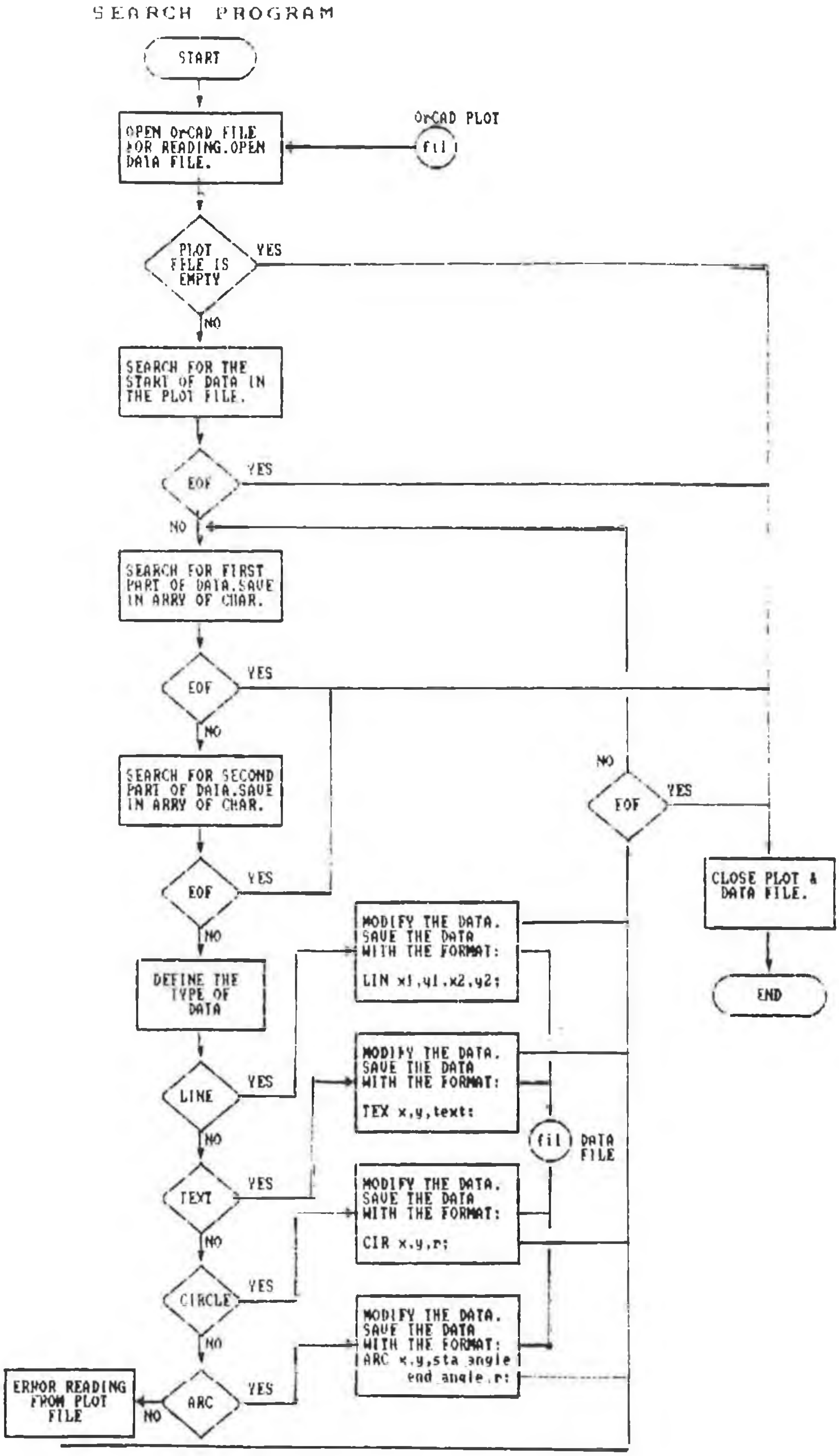


fig. 6-2. The Flowchart of the Search Program

"ERROR: OPEN PLOT FILE". Then it will close the plot and data files and exit.

But if the plot file contains the graphical information, the Search Program will start searching for the start point of the data, which is for HP-format, the term (SP1).

After finding (SP1) inside the plot file, the Search Program checks the plot file to ensure that it hasn't reached its end.

If the Search Program encountered (EOF) inside the plot file it will close the plot and the data files, and exit.

This check for the end of file will be accomplished after every search for new data in the plot file.

Now if it is not the end of the file, the Search Program will start to search for the first part of the graphical data, which always has one of the following formats:

1. PA(x,y); (the normal case).
2. AA(x,y),+/- start angle, the arc index; (in the case of drawing a chain of arcs).

Then it will save the captured data in an array of character (First). After that the Search Program makes the end of file check on the plot file. If it is passed, the Search Program will start searching for the second part of the graphical data, which will take one of the following formats:

(1) PA(x,y) for drawing line.

(2) LB text for writing a text.

(3) CI r for drawing a circle.

(4) AA(x,y),+/- start angle, the arc index for drawing an arc.

Then it will save the captured data in another array of character (Second). Also the Search Program will check for (EOF), after the previous search.

Now the Search Program has a complete information for drawing a line, a circle, an arc, or writing a text. To define the type of the captured data the Search Program scans the first two characters in the array (Second). The first two characters in the second part of the data, always defines the type of graphical information.

Then the Search Program will save these two characters in character array (Comp). After that, the Search Program begins to extract the main information about the captured data (dimensions, coordinates, text, angles). Then saves them in two arrays of character ((C) contains the extracted information from the first part of graphical data, (D) contains the extracted information from the second part of graphical data).

By using strcmp() function in TURBO. C. the Search Program will compare between the content of (Comp) and a pointer to options, (contains the first two characters to every expected graphical entities, PA,LB,CI,AA).

If the content of (Comp) array was "PA" this will indicate to a line command in the HP-formated plot file. In this case, the Search Program will do the suitable manipulation and modification to the format of the captured data to put it in the suitable format for the line() function in TURBO. C. (the new formated data: LIN x1,y1,x2,y2;). Then the Search Program will save the new formated data in the data file.

If the content of (Comp) array was "LB", this will indicate to a text command in the HP-formatted plot file. Also in this case, the suitable manipulation and modification will take part to put the captured data in the suitable format for the text() function in TURBO. C. (the new formated data: TEX x,y,*text*;).

Then the Search Program will save the new formated data in the data file.

If the content of (Comp) array is "CI", this will signal to a circle command in the HP-formatted plot file. Also the suitable manipulation and modification will take a part to put the captured data in the suitable format for the circle() function in TURBO. C. (the new formatted data: CI x,y,*r*;). Then the Search Program will save the new formatted data in the data file.

If the content of (Comp) array is "AA", this will signal to an arc command in the HP-formatted plot file. In this case, some difficult work has carried out to modify the captured data, to put it in the suitable format for the arc() function in TURBO. C..

The arc command in HP-formatted plot file has two formats:

(1) PA(x1,y1);AA(x,y),+/- start angle,the arc index;

(2) AA(x1,y1),+/- start angle,the arc index;AA(x,y),+/-start angle,the arc index.

The arc() function in TURBO.C. has the format:

arc(x,y,start angle,end angle,r)

The first format of the arc command, contains the start point of the arc point PA(x1,y1), the centre of the current arc point(x,y), the direction of the arc, the start angle, and the arc index. The second format of the arc command, contains the centre of the previous arc, point AA(x1,y1) (no matter what the other information inside the first part of data, in this case). The second part of data contains the centre of the current arc point(x,y), the direction of the arc, the start angle, and the arc index.

While the arc() format contains the centre of the arc point(x,y), the start angle, the end angle and the radius *r*.

There are two ways, to calculate the end angle of the arc from the components of the arc command format. If the arc index was equal

(30):

the end angle = +/- start angle + 180

But if the arc index was equal (15):

the end angle = +/- start angle + 90

To calculate the radius of the arc, from the components of the arc command:

xr = x1 - x

yr = y1 - y

for the first format (of the arc command):

r = sqrt(pow(x,2.0) + pow(y,2.0))

for the second format (of the arc command):

r = (1.0/2.0) * (sqrt(pow(x,2.0) + pow(y,2.0)))

The new formatted data: ARC x,y,start angle, end angle, r

Then the Search Program will save the new formatted data in the data file. If the content of (Comp), wasn't one of the previous characters, the Search Program will print on the screen:

"ERROR: READING FROM PLOT FILE".

The three characters in the beginning of every new formated data, was added to simplify the recognition of the data type, in the second part of the software.

In the end of this search, the Search Program will make an (EOF) test. If the Search Program encounters (EOF) inside the plot file it will close the plot and the data files, and exit.

If the Search Program doesn't encounter (EOF), it will return and repeat the previous work. So it will search for the first and second parts of graphical information, then it will manipulate, modify them, and save the new formatted data in the data file.

The data is organized in lines inside the data file, each line contains information for drawing one graphic entity:

```
LIN x1,y1,x2,y2;
TEX x,y,text;
CIR x,y,r;
ARC x,y,start angle,end angle,r;
```

The work of the Sketch Program is to read the new formatted data from the data file, and prepare it so that it can be drawn on the screen. The Sketch Program will draw the schematic (Sketch1) or the layout diagram (Sketch2), when it is called from inside EXP-Test System. The difference between Sketch1 and Sketch2 is that, Sketch1 reads the data file, which contains the graphical information for drawing the schematic diagram, while Sketch2 reads the data file, which contains the graphical information for drawing the layout diagram. Fig. 6-3 shows the flow chart of the Sketch Program.

In the beginning the Sketch Program initializes the graphics system and puts the system in the graphics mode. Then it selects the style of the line, which will be used for writing the text. Also it will set the color of the current drawing and the current background color. After that the Sketch Program will open the data file and read it. Then it will check the data file to ensure that it is not empty. The Sketch Program will print on the screen:

"ERROR: OPEN FILE", and exit, if the data file is empty.

If the data file isn't empty , the Sketch Program will go to the next stage. In the next stage the Sketch Program will scan the first three characters of the first line in the data file, and save them in character array (Comp). Also by using strcmp() function, in TURBO. C., the Sketch Program will compare between the content of (Comp) and a pointer to options, (which contains the different types of the first three characters for every line in the data file).

SKETCH THE SCHEMATIC DIAGRAM

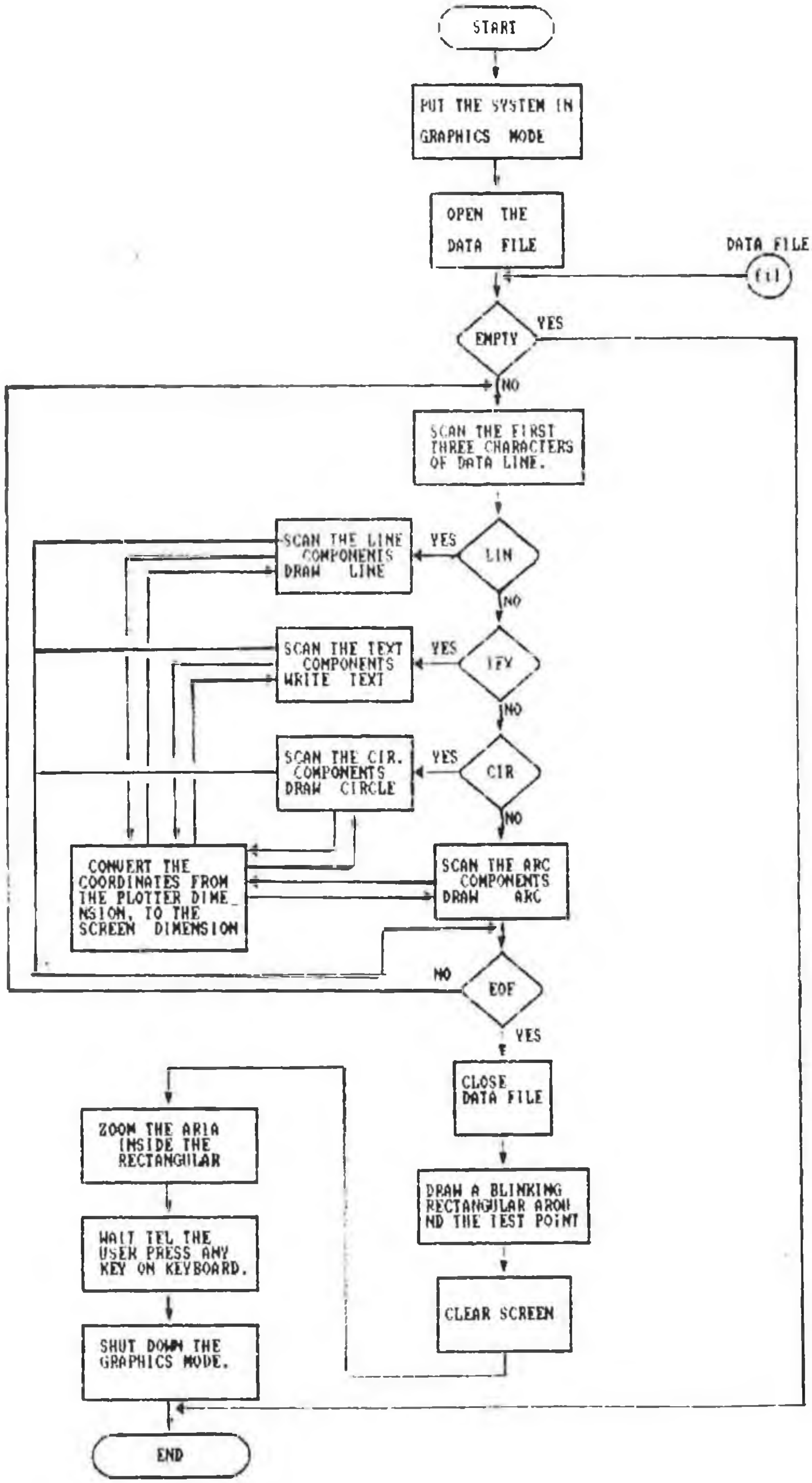


fig. 6-3. The Flowchart of the Sketch Program

If the content of (Comp) array is "LIN", this will indicate that the data file, contains data for drawing a line. The Sketch Program will scan the rest of the current data line put the graphics information in array of character (Temp). Then the Sketch Program scans (Temp), separates its content to: x1,y1,x2,y2. By using convert() function, which converts the coordinates from the plotter scale to the screen scale, and using line() function, the Sketch Program will draw a line on the screen.

If the content of (Comp) array is (TEX), (CIR) OR (ARC), the same procedure will take place. But outtextxy() function, will be used for writing the specified text, circle(), will be used for drawing a circle, and arc() function, will be used for drawing an arc.

Then the Sketch Program will make the end of file test, to ensure that the file hasn't reached the end.

If the Sketch Program pass this test, it will return back to scan another line of data inside the data file. But if it didn't pass the end of file test, the Sketch Program will close the data file.

At this stage, a complete schematic will appear on the screen. So the Sketch Program will draw a blinking rectangle around the test point, to draw the attention of the user to the wanted place.

This blinking rectangle will continue for 20 seconds, then the Sketch Program will clear the screen, and will zoom onto the area inside the rectangular. Also the test point will be shown blinking on the screen to notify the user of the correct position of the test point.

After a while, in addition to the schematic diagram, a "PRESS ANY KEY TO CONTINUE" sentence will appear on the screen, to force The Sketch Program to wait until the user presses any key on the keyboard.

If the user has pressed any key, the Sketch Program will shut down the graphics mode and exit.

The convert function consists of two functions. The first one is x_conv() function, which convert (x) values from the plotter scale to the screen scale. The second function is y_conv() function, which convert (y) values from the plotter scale to the screen scale. In the beginning when the plot file is created, the size of the paper, which the plotter will plot the schematic on, is defined to (A4). The location of the coordinate origin (0,0 plotter units) and the orientation of the X-and Y- axis is shown in fig. 6-4.

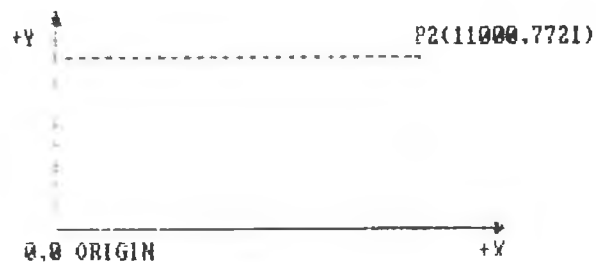


fig. 6-4. The Location of the (0,0 Plotter Units)

The (A4) size paper area is divided to (11000) in X-direction and to (7721) in Y-direction.

The initialize of the graphic system defines the screen to be divided to (320) in X-direction and to (200) in Y-direction.

Fig. 6-5 shows the location of the coordinate origin (0,0 screen units) and the orientation of the X-and Y- axis.

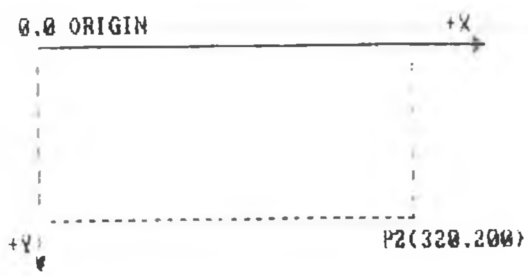


fig. 6-5. The Location of the (0,0 Screen Units)

The x_conv() function, multiply the (x) value, which originally come from the HP-formatted plot file (the coordinates shown in fig. 6-4), by x_conversion rate:

x_conversion rate = 320 / 11000 = 0.029

(x) value (in screen scale) = x_conversion rate * (x) value (in plotter scale)

Then x_conv() function, returns the (x) value in the screen scale.

Because the orientation of Y-axis in the plotter scale is opposite that of the orientation of Y-axis in the screen scale.

The y_conv() function, subtracts the (y) value, which originally come from the HP-formatted plot file, from the maximum value to (y) in the plotter scale (7721). Then multiply the result by y_conversion rate:

y_conversion rate = 200 / 7721 = 0.026

(y) value (in screen scale) = y_conversion rate * (7721 - (y) value (in plotter scale)

Then y_conv() function, returns the (y) value in the screen scale.

6.4. Integrating OrCAD/SDT III System with EXP-Test System

Now the Sketch program (Sketch1 and Sketch2) is ready for drawing the schematic or the layout diagram. To do that in conjunction with EXP-Test System, the CALL clause, from inside VP-Expert, should be used. The CALL clause, should be added to the IF-THEN rules, in the suitable place to increase the competence of EXP-Test System. As shown in the following example:

```
RULE    POWER_SUPPLY
IF      CHECK_TRANSFORMER = YES
THEN    CALL \TC2\SKETCH1
        CALL \TC2\SKETCH2
        CALL \TC2\EXP_T1
        LOADFACTS  FILE9
        FIND THE_TEST
        THE_PROBLEM = FOUND;
```

In response to the previous rule, EXP-Test System will display:

TO CHECK THE TRANSFORMER:

DISCONNECT THE TRANSFORMER'S OUTPUT FROM (J9).

THEN PUT THE OSCILLOSCOPE'S PROBE AT POINT (1).

TO LOCATE POINT (1) AND CONTINUE THE TEST, CHOOSE YES, AND PRESS <ENTER>. OTHERWISE CHOOSE NO, AND PRESS <ENTER>.

YES NO

when POWER_SUPPLY rule is fired (by choosing YES), the inference engine will call first of all the SKETCH1 file, which draws the schematic diagram of the circuit, then the inference engine calls the SKETCH2 file, which draws the layout diagram of the circuit. The user should put the probe at point (1), then he should press any key on the keyboard to continue testing the broken_down machine.

CHAPTER 7

IMPLEMENTATION

7.1. Introduction

EXP-Test System was developed as a system for electronic fault diagnosis. It has an advantage through the inclusion of both types of knowledge, shallow and deep knowledge, (The shallow knowledge is the knowledge about fault diagnosis and the deep knowledge is the knowledge about electronic circuitry).

The shallow knowledge in EXP-Test System was represented in a number of the IF-THEN rules. These IF-THEN rules give the system an easy way of representing the domain knowledge of the expert in the maintenance, also it gives the system a high speed execution with the ability to extend the system easily, by adding some rules.

The deep knowledge is included in the union of the Graphics facility, and the intelligence of the technician.

Diagnosis is intrinsically a goal directed reasoning process; a goal being to prove that a particular module or component in a circuit is faulty [63].

The procedure for testing the plotter's circuitry, in EXP-Test System, was defined in cooperation with the maintenance expert in the school of electronic engineering at Dublin City University.

As mentioned, different experts use different strategies for solving electronic faults.

In NODAL, the expert system which was developed in Trinity College Dublin for testing the switching mode power supply, different strategies were used for electronic fault diagnosis.

"The model within NODAL is hierarchical containing at least two levels; a module level and a component level. If the modules are large or complex then they may be subdivided into sub-modules in the model" [63].

"At a module level the fundamental principle is that if a module has good signals at its input and a bad signal at one of its outputs then it is faulty" [35].

The strategy, which EXP-Test System uses to trace electronic faults, depends on regressing along a faulty line. This is very similar to that which the technician does in the diagnosis of electronic faults. So for example a 'bad' signal is found at the (5V) output of the power supply, the EXP-Test System will trace all the signals on that line, with the intention of defining the damaged components.

7.2. U.U.T. description

As mentioned, EXP-Test System was developed for electronic fault diagnosis in CNC machines. Because of the unavailability of a CNC machine in the electronic school, the HP 7475A Graphics plotter has been used as U.U.T. (unit under test). The HP 7475A Graphics plotter has the same principle of the CNC machines.

Fig. 7-1, shows the block diagram of HP 7475A Graphics plotter.

The HP 7475A Graphics plotter uses microprocessor-based logic to convert digital instructions into a graphic plot. The microprocessor receives instructions from either an internal ROM (read only memory) program or an external controller through the I/O (input/output) circuits. It then issues data to the pen drive and paper drive motor servo systems and the pen down circuit to produce the plot.

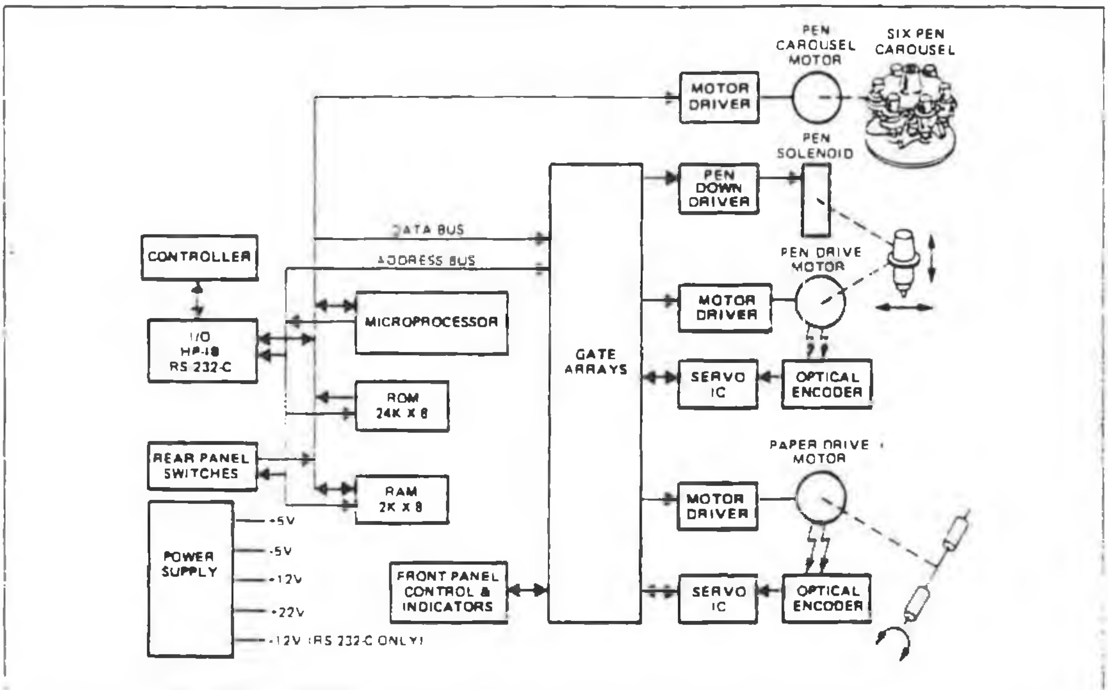


fig. 7-1

7475A Simplified Block Diagram

The two drive motors are reversible dc motors. Encoders on each motor transmit rotation data back to their respective servo IC. One motor drives grit wheels which moves the plotting medium, while the other motor moves the pen across the plotting surface.

The most of the plotter's faults, usually happen in the power supply, motors and in the motor servo systems. Because of that, EXP-Test System attempted, in its electronic fault diagnosis, to cope with all these faults, and define the damaged components inside these units.

Two identical servo systems are used to drive the motors.

Fig. 7-2, shows the block diagram of a motor servo system.

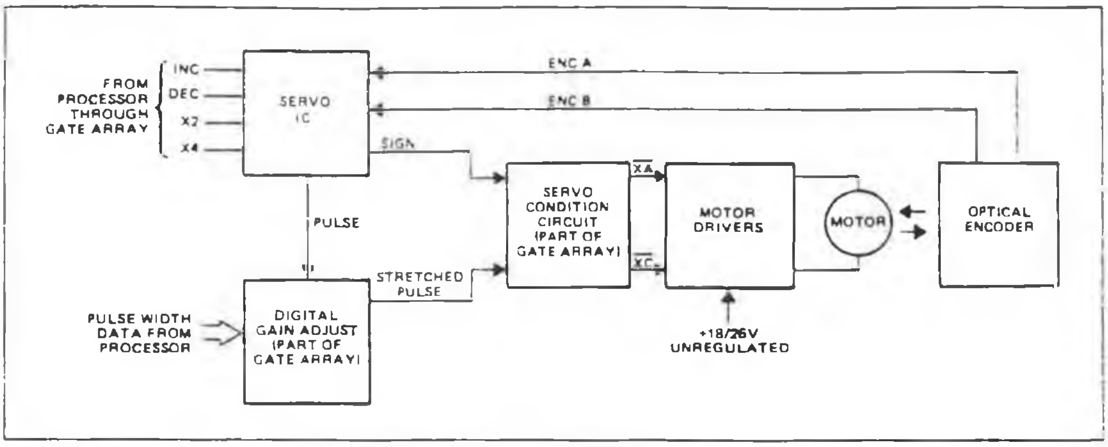


fig. 7-2

Motor Servo System Block Diagram

Digital move commands are generated and sent by the microprocessor through the gate arrays to the servo chips. The servo chips provide the interface to the microprocessor, decodes the encoder signals, sums position errors, estimates velocity and sums it, and transforms the servo error to a pulse-width-modulated output. The servo chips output the pulse_width_modulated signal back to the gate array where the motor drive pulses are stretched to the proper width in the digital gain circuit. The servo conditioning circuit, also in the gate array, then passes the pulses on to the motor drivers through either the XA or XC line, depending on the direction of rotation indicated by the sign voltage from the servo chip. As the mechanical system moves, optical encoders mounted on the shaft of each motor send back digital pulses to the servo chip to close the servo loop.

To maintain a consistent and predictable movement, it is essential to control the amount of power applied to the motor by each pulse.

The pulse amplitude depends on the actual voltage output of the motor drive power supply. The pulse width is modified to compensate for pulse amplitude so that the pulse represents the proper amount of power.

Five major voltages are generated by the 7475A circuitry. Low current linear supplies provide the +12V and -5V required for the servo IC's. The +5V supply provides the power for the remaining logic circuitry. Regulation for the +5V linear supply is provided. The unregulated 18/26V supplied to the main drive motors and the pen carousel drive motor. The fifth supply is the -12V source used in conjunction with the +12V supply to operate the I/O (input/output) line drivers.

Fig. 7-3 shows the 7475A Graphics plotter's power supply.

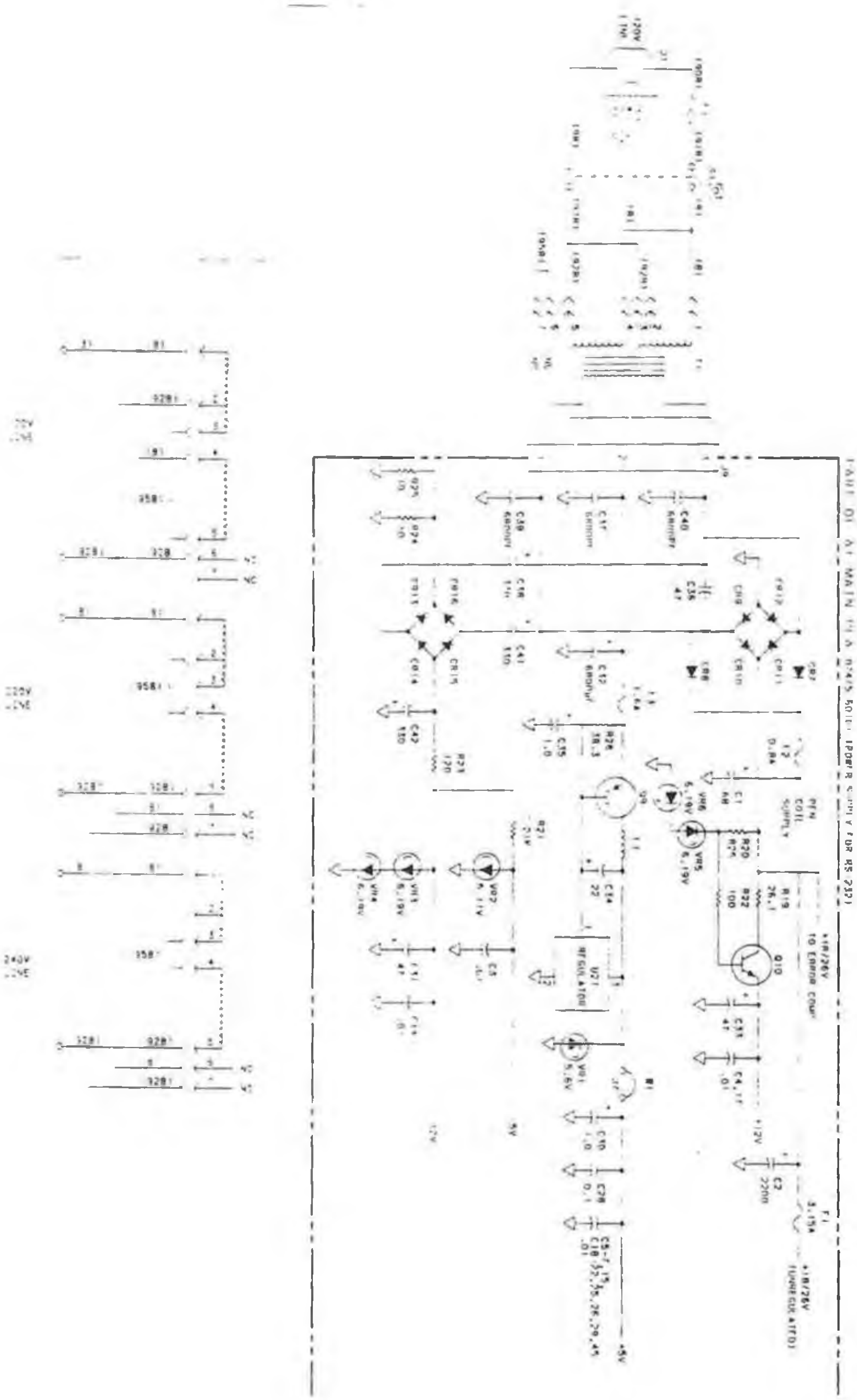


fig. 7-3. The 7475A Graphics Plotter's Power Supply

7.3. Approach Adopted

The development of the EXP-Test System proceeded in two stages. The First stage was to study the domain, which was the design of the 7475A Graphics plotter, its construction and the problems which might be encountered during it's use. A deep study to the 7475A Graphics plotter's circuitry was done.

The contact with the manufacturer's service board, concluded by a list of some previous faults, which happened during the history of the plotter.

The second phase of the project, which required much more original thought and work, was to design and built the supporting programs (in TURBO.C.) and the final expert system with VP-Expert. The design structure imposed by VP-Expert meant that, in general, the system would have knowledge about what should be happening if the plotter were in perfect working order. So the system looks for what should be there and when there is a discrepancy between the expected behaviour and the actual behaviour of the plotter, it tries to localize the source of this discrepancy. When the system has done this it then proposes a hypothesis about the cause of the fault based on the location of the discrepancy.

Therefore the fault is diagnosed not by what is happening, but by what is not happening in the plotter's circuitry.

7.4. Design and Development

7.4.1. Introduction

Once VP-Expert system was adopted as the shell for EXP-Test System, the appearance of the system to the user and the style of consultation had also been decided as VP-Expert system dictates this to the builder of the expert system. This is due to the rigid question and answer framework which VP-Expert system provides as its only means of communication with the user.

VP-Expert system was selected as the development environment for EXP-Test System, because it was the only expert system shell available in the electronic school at the time of implementation.

VP-Expert system has special features including:

(1) An "inference engine" that uses backward and forward chaining for problem solving.

(2) Optional development windows that let the user observe the behind-the-scenes path of the inference engine as it navigates the knowledge base to solve problems during a consultation.

(3) Confidence factors that let the user account for uncertain information in a knowledge base.

(4) Simple English rule construction.

(5) The ability to explain its actions during a consultation.

(6) Knowledge base "chaining", which lets the user create knowledge bases that would otherwise be too large to fit in memory.

(7) A built in text Editor.

(8) Automatic question generation.

(9) The ability to record and graphically display the rule-by-rule search pattern used behind the scenes during a consultation.

(10) Rapid execution of the knowledge base.

(11) The ability to execute external DOS programs.

(12) Floating point math functions. [66]

One of the main problems at the outset of design and development was the complete lack of any examples, or previously implemented systems either in EMYCIN or in any similar expert system shell. Although there is a large amount of information in papers and books about expert systems, there is very little information available anywhere on actual design and implementation of working systems [64].

7.4.2. Structure

The goal of EXP-Test System throughout are to detect 'faults' and find 'solutions'. To approach these goals and build the automatic test system, supporting software has been developed in TURBO.C. with EXP-Test System.

This software reduces the dependency on the user of EXP-Test System in defining the quality of the signals.

In NODAL it is very important that the user give help to the system, in defining the fault. NODAL usually asks the user about the nature of the signals at various points in the power supply circuit.

"In order to prove this the clauses in the rule premise have to be proved true. The information that the module is of type one-one is available in the module frame. The user will have to be asked for the other information. The output is found to be connected to Node-12 so the user is asked for the quality of the signal at Node-12 and at A to which the input is connected. If the signal on the output is bad and the signal on the input is good then the module is concluded to be faulty.

Setup for Test Vector 1

What is the SIGNAL of A? Good

What is the SIGNAL of B? Good

What is the SIGNAL of NODE-12? Good"

[63]

EXP-Test System usually, asks the user to move the probe from one test point to another, and it will ask him for some extra information about the quality of the signals in a few places.

Fig. 7-4 shows the complete software, which was developed to achieve the project's goals.

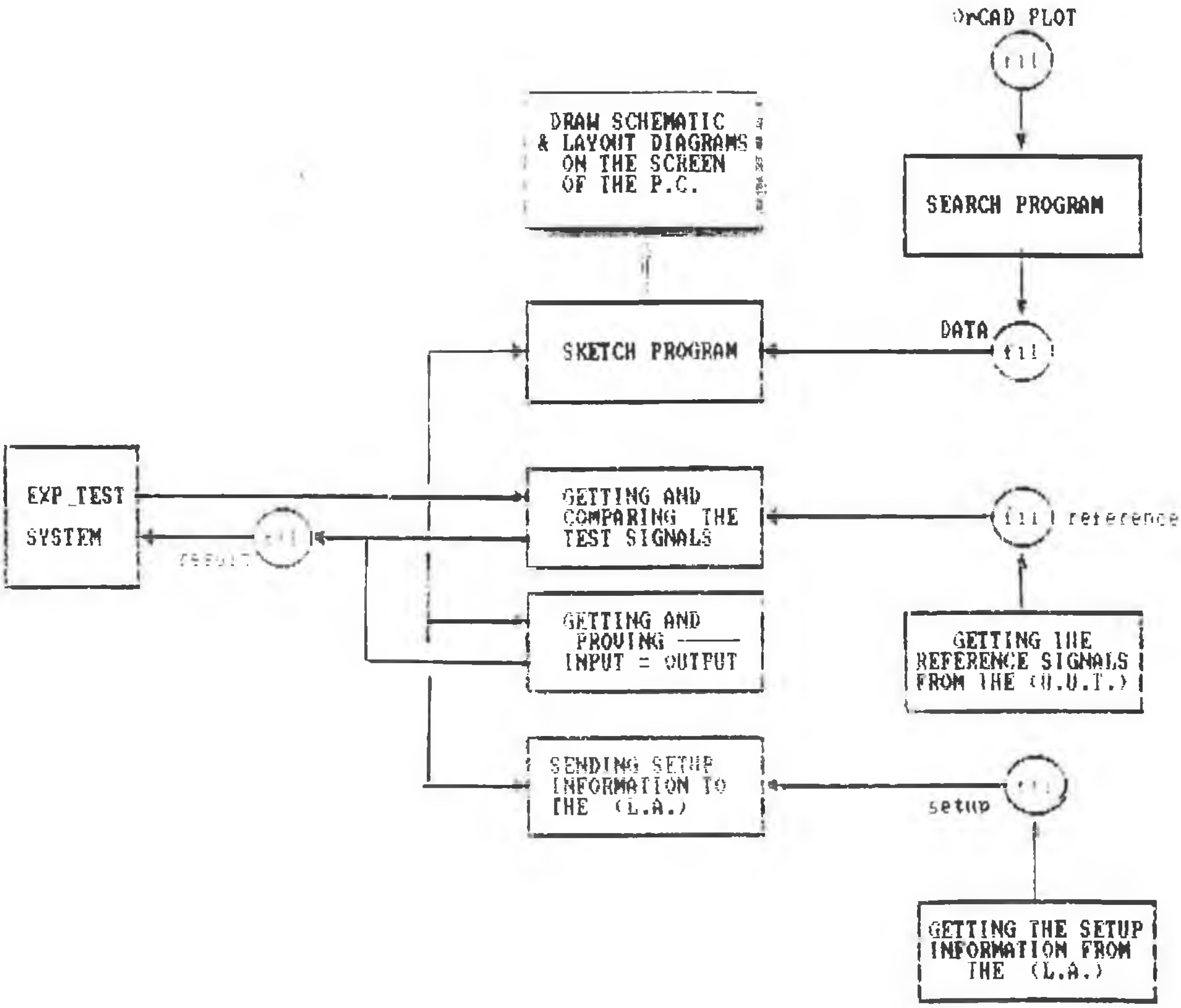


fig. 7-4. The Complete Software Configuration

First of all, EXP-Test System asks the user to check the main power supply. This work should be done before testing any point in the plotter's circuitry.

After that EXP-Test System will ask the user to put the probe of the oscilloscope (for testing the analogue signals in the plotter's circuitry) or of the logic analyzer (for testing the digital signals in the plotter's circuitry), on one of the test points.

Immediately after that EXP-Test System will call SKETCH program, which will draw the schematic and the layout diagrams on the screen.

EXP-Test System will ask the user now if he has put the probe at the wanted test point. When the user choose YES from the choice list, EXP-Test System will call the Getting and Comparing the Test signals program.

The work of this program is to get the signal from the test point and compare it with the reference signal. Then calculating the difference and the allowable tolerance and save them in a text file called RESULT.

EXP-Test System will take these two values from RESULT, and will compare them to define the problem and the solution.

In the case of using the logic analyzer to get the test signals. EXP-Test System will call the Sending Setup Information program, which gets the setup information from the setup file and sends it to the logic analyzer.

EXP-Test System will call Getting and Proving input = $\overline{\text{output}}$ program for testing the condition of the inverters in the motor servo systems.

To test the output of the encoders (the feedback signals), EXP-Test System will ask the user to put the oscilloscope's probe at the encoder's output and define the quality of the signals. This is the only case which EXP-Test System asks the user for the quality of the signal. That is because of the confusion which the feedback causes during testing the motor servo systems. The motor usually stops in the case of any fault in the motor servo system, that will cause the feedback signal to vanish. So if EXP-Test System calls Getting and Comparing the Test Signals program, to compare the feedback signal with a reference signal it will conclude that the problem is always in the encoder, whereas it might be in any part of the motor servo system.

7.4.2.1. Supporting Software

The supporting software was developed to enhance the automatic work of EXP-Test System. It is divided into six different types, each type supports EXP-Test System in one side of the automatic test. A brief explanation on each type will be mentioned. These types are:

(1) Search and Sketch programs

An expanded explanation has been already given about these two programs which support the graphics facility accompanying EXP-Test System.

(2) Getting the Reference Signals from the (U.U.T.)

This software was divided into two programs. The first one, get the reference analogue signals from (U.U.T.) through the IEEE Interface Card by using the oscilloscope. This program will convert the captured data (the raw data in ASCII code) to a useful interpretation. After that it will save the processed data in a text file, called REFERENCE.

The second one, get the reference digital signals from

(U.U.T.) through the IEEE Interface Card by using the Logic Analyzer. Then this program will convert the captured data (the raw data in ASCII code) to a useful interpretation. After that it will save the processed data in a text file also, called REFERENCE.

(3) Getting and Comparing the Test Signals

This software was divided into two parts. The first part gets the test analogue signals from (U.U.T.) through the IEEE Interface Card by using the oscilloscope. Then this program will convert the captured data (the raw data in ASCII code) to a useful interpretation. After that it will call the reference file, and compare between the test and the reference signals to calculate the difference. In the end it will define the allowable tolerance and save it with the difference in a text file called RESULT.

The second part, get the test digital signals from (U.U.T.) through the IEEE Interface Card by using the Logic Analyzer. Then this program will convert the captured data (the raw data in ASCII code) to a useful interpretation. After that it will call the reference file, and compare between the test and the reference signals to calculate the sum of the total difference, which equals zero if the plotter is in good condition. In the end it will save the sum of the total difference value in a text file called RESULT.

(4) Getting and Proving Input = Output

This program will get the test digital signals from (U.U.T.) through the IEEE Interface Card by using the Logic Analyzer. The input and the output signals of the inverter can be captured, by using the multi-channels pod of the Logic Analyzer. Then this program will convert the captured data (the raw data in ASCII code) to a useful interpretation. After that it will check that the input signal is opposite to the output signal (to prove that the inverter is ok), and save the result in RESULT.

(5) Getting the Setup Information from the Logic Analyzer

This program will just get the setup information from the Logic Analyzer and save it in a setup file.

(6) Sending the Setup Information to the Logic Analyzer

This program will call the setup file and send the setup information to the Logic Analyzer.

The flow charts of all the previous mentioned programs are in appendix A.

7.4.2.2. Designing the Rule Set

Developing the rules to be used by EXP-Test System was the largest development task representing approximately four months work which formed the core of the project. The result is about one hundred rules. A fuller description is provided in section 4.10.1. A typical rule from this expert system is shown here:

```
RULE POWER_SUPPLY12
IF CHECK_PEN_SUPPLY = YES
THEN CALL\TC2\SKETCH1
     CALL\TC2\SKETCH2
     CALL\TC2\EXP_T10
     LOADFACTS  FILE9
     FIND THE_PEN_SUPPLY
     THE_12V_TEST = FOUND;
```

```
RULE POWER_SUPPLY13
IF delta1 >= (stander1)
THEN WOPEN 4,15,1,8,60,14
     ACTIVE 4
     DISPLAY
   "THERE IS  A  PROBLEM  IN  THE  PEN  COIL SUPPLY'S OUTPUT, TO
    DEFINE WHICH COMPONENTS CAUSE  THE  PROBLEM, YOU SHOULD MAKE
    TEST TO ANOTHER POINT. <PRESS ANY KEY> TO CONTINUE.~"
     WCLOSE 4
     RESET delta1
     RESET stander1
     THE_PEN_SUPPLY = FOUND
     FIND MOR_PEN_SUPPLY
ELSE CLS
     DISPLAY
    "THE PEN  COIL  SUPPLY  IS OK, <PRESS ANY KEY>  TO  CONTINUE
     TESTING THIS DEVICE.~"
     RESET delta1
     RESET stander1;
```

7.4.2.3. Domain expertise

There were two clear stages in developing the rule set.

First there was the task of developing "domain expertise" in other words becoming familiar with various aspects of the plotter's electronic circuitry.

It was important to understand the actual construction of the plotter's circuitry as will as understanding the circuitry's working at a logical and digital level.

This included a wide knowledge about the problems and the components which would be most likely to be at fault.

A great deal of the information or "knowledge" gathered at this stage of the project seemed extremely difficult to put into rule format. Details of the particular things which could be wrong with the plotter, and how they might be checked, confirmed and fixed.

The second stage of the development consisted of organizing this information and constructing rules from it.

7.4.2.4. Knowledge Engineering

One approach to design rules is by attempting to construct consultation "trees" on paper, by writing out lists of useful questions which could be asked, and linking them with arrows representing the order in which they should be presented to the user so that they would be most intelligible to him. A rule with two possible answers (eg. YES or NO), would have two arrows from it, for the two possible answers. These arrows would go to more questions, or if enough information had been collected then, they would point to a fault and associated solution.

The approach which used to design rules in this project was by drawing the initiative flow chart of EXP-Test System. This flow chart was containing the useful questions which could be asked to the user and the expected answers.

This was found to be a very useful approach for developing the overall structure and also for visualizing how the system would appear to the user.

The first stage of the knowledge engineering process was to develop the overall structure of the system, this was done along conventional knowledge engineering lines.

The localisation technique is very similar to that which an expert would be expected to take.

7.5. Certainty Factors

EXP-Test System didn't use certainty factors in its rules. The reasons for this are very interesting and are a reflection on the fault domain. In most instances the system asks the user to move the probe from one test point to another, or if there is a signal, in some test point (feedback signals), this is a simple YES/NO query where there is very little vagueness. There is a signal or there is not! If the expected signal is not there, then there is a fault or problem in the circuit which produces or uses that signal, so the system localizes to that part of the circuit and tests another point or section of the circuit.

There is very little room for uncertainty in electronic circuits, especially in digital circuits. A line is generally either high, low or carrying a signal! The nature of digital test equipment also leads to precise results of tests, the user either sees a signal on an oscilloscope screen or he doesn't. This is very different to the situation in medical diagnosis where estimates and qualitative judgements must be made without clear quantitative results [64].

7.6. EXP-Test System in Operation

To test the performance of EXP-Test System and to improve its ability in electronic fault diagnosis, some experiences were applied on it. To do that, wires were connected from the test points (A,B,C,D,E,F,G,H,I,J,K,L,M) on the motor servo systems, to an external breadboard to simplify getting the test signals from that points. But it was very difficult to reach the defined test points inside the plotter's power supply, and changing the signal's characteristics to simulate the faults. So a function generator and a power supply was used to simulate the signals of the plotter's power supply.

The function generator's signals were used to simulate the transformer's output (to give 10.5 ACV).

The power supply's signals were used to simulate all the continuous voltages appearing on the plotter's power supply board.

So by using the front panels of the function generator and the power supply, the amplitude of the signals can be changed (increase or decrease the volt values) to simulate the fault cases.

The following examples will explain how EXP-Test System diagnoses the electronic faults to achieve the solution:

The first fault was simulated by decreasing (-5V) DCV, and making it less than the allowable tolerance. The plotter's power supply fails test POWER_SUPPLY4.

The dialogue was as follows (the underlined text is the user choice):

WELCOME TO

EXP-Test SYSTEM

SYSTEM FOR TESTING CNC MACHINE

<PRESS ANY KEY TO CONTINUE>

(The user should press any key)

CHECK THE MAIN POWER PLUG (220) ACV.

IF THE MAIN POWER IS OFF, (MOVE THE CURSOR TO OFF AND PRESS <ENTER>, IF ON PRESS <ENTER>.

ON OFF

TO CHECK THE TRANSFORMER:

DISCONNECT THE TRANSFORMER'S OUTPUT FROM (J9).

THEN PUT THE OSCILLOSCOPE'S PROBE AT POINT (1).

TO LOCATE POINT (1) AND CONTINUE THE TEST, CHOOSE YES, AND PRESS <ENTER>. OTHERWISE CHOOSE NO, AND PRESS <ENTER>.

<u>YES</u> NO

THE TRANSFORMER IS OK, <PRESS ANY KEY> TO CONTINUE TESTING THIS DEVICE.

(The user should press any key)

PUT THE OSCILLOSCOPE'S PROBE AT POINT (2),

IF YOU ARE READY, CHOOSE YES, AND PRESS <ENTER>.

OTHERWISE CHOOSE NO, AND PRESS <ENTER>.

<u>YES</u> NO

THE (5V) OUTPUT IS OK, <PRESS ANY KEY>, TO CONTINUE TESTING THIS DEVICE.

(The user should press any key)

PUT THE OSCILLOSCOPE'S PROBE AT POINT (5).

IF YOU ARE READY, CHOOSE YES, AND PRESS <ENTER>.

OTHERWISE CHOOSE NO, AND PRESS <ENTER>.

<u>YES</u> NO

THE (-12V) OUTPUT IS OK, <PRESS ANY KEY>, TO CONTINUE TESTING THIS DEVICE.

(The user should press any key)

PUT THE OSCILLOSCOPE'S PROBE AT POINT (7).

IF YOU ARE READY, CHOOSE YES, AND PRESS <ENTER>.

OTHERWISE CHOOSE NO, AND PRESS <ENTER>.

YES NO

THE PROBLEM IS DEFINED, THE EXPECTED DAMAGED COMPONENTS ARE:

(1) C3.

(2) VR2

PRESS <ENTER>, THEN <Q>, TO EXIT.

The EXP-Test System has defined the fault candidate list, which is C3, VR2. All what is wanted from the user is to test these components and change the faulty one.

The previous test was one of many tests, which applied on the EXP-Test System to improve its capability in troubleshooting faults in the plotter's power supply.

The second fault was simulated by getting the input (point A) and the output (point E) signals of the inverter (paper motor drivers) from the input only. This will indicate that the paper motor drivers, which invert the input signals, is faulty. This is because the input and the output signals will be the same (two signals from the same source).

This example will show the capability of EXP-Test System in defining the damaged components in the digital circuits (motor servo system). The motor servo system fails test P-D-MOTOR5. The dialogue was as follow (the underlined text is the user choice):

WELCOME TO

EXP-Test SYSTEM

SYSTEM FOR TESTING CNC MACHINE

<PRESS ANY KEY TO CONTINUE>

(The user should press any key)

CHECK THE MAIN POWER PLUG (220) ACV.

IF THE MAIN POWER IS OFF, (MOVE THE CURSOR TO OFF AND PRESS <ENTER>, IF ON PRESS <ENTER>.

ON OFF

TO CHECK THE TRANSFORMER:

DISCONNECT THE TRANSFORMER'S OUTPUT FROM (J9).

THEN PUT THE OSCILLOSCOPE'S PROBE AT POINT (1).

TO LOCATE POINT (1) AND CONTINUE THE TEST, CHOOSE YES, AND PRESS <ENTER>. OTHERWISE CHOOSE NO, AND PRESS <ENTER>.

YES NO

THE TRANSFORMER IS OK, <PRESS ANY KEY> TO CONTINUE TESTING THIS DEVICE.

(The user should press any key)

PUT THE OSCILLOSCOPE'S PROBE AT POINT (2),

IF YOU ARE READY, CHOOSE YES, AND PRESS <ENTER>.

OTHERWISE CHOOSE NO, AND PRESS <ENTER>.

YES NO

THE (5V) OUTPUT IS OK, <PRESS ANY KEY>, TO CONTINUE TESTING THIS DEVICE.

(The user should press any key)

PUT THE OSCILLOSCOPE'S PROBE AT POINT (5).

IF YOU ARE READY, CHOOSE YES, AND PRESS <ENTER>.

OTHERWISE CHOOSE NO, AND PRESS <ENTER>.

YES NO

THE (-12V) OUTPUT IS OK, <PRESS ANY KEY>, TO CONTINUE TESTING THIS DEVICE.

(The user should press any key)

PUT THE OSCILLOSCOPE'S PROBE AT POINT (7).

IF YOU ARE READY, CHOOSE YES, AND PRESS <ENTER>.

OTHERWISE CHOOSE NO, AND PRESS <ENTER>.

YES NO

THE (-5V) OUTPUT IS OK, <PRESS ANY KEY>, TO CONTINUE TESTING THIS DEVICE.

(The user should press any key)

PUT THE OSCILLOSCOPE'S PROBE AT POINT (8).

IF YOU ARE READY, CHOOSE YES, AND PRESS <ENTER>.

OTHERWISE CHOOSE NO, AND PRESS <ENTER>.

YES NO

THE (+18/26V) OUTPUT IS OK, <PRESS ANY KEY>, TO CONTINUE TESTING THIS DEVICE.

(The user should press any key)

PUT THE OSCILLOSCOPE'S PROBE AT POINT (11).

IF YOU ARE READY, CHOOSE YES, AND PRESS <ENTER>.

OTHERWISE CHOOSE NO, AND PRESS <ENTER>.

YES NO

THE (+12V) OUTPUT IS OK, <PRESS ANY KEY>, TO CONTINUE TESTING THIS DEVICE.

(The user should press any key)

PUT THE OSCILLOSCOPE'S PROBE AT POINT (12).

IF YOU ARE READY, CHOOSE YES, AND PRESS <ENTER>.

OTHERWISE CHOOSE NO, AND PRESS <ENTER>.

YES NO

THE (PEN COIL SUPPLY) OUTPUT IS OK, <PRESS ANY KEY>, TO CONTINUE TESTING THIS DEVICE.

(The user should press any key)

SWITCH THE PLOTTER (U.U.T.) OFF, PUT A BLANK PAPER, ONE PEN AT LEAST, (NO1). THEN PRESS (P1) & (P2), AT THE SAME TIME SWITCH ON THE PLOTTER. THE TEST PROGRAM WILL RUN, TESTING THE OPERATION OF EVERY MOTOR, THAT CALLED (THE INTERNAL TEST PROGRAM). BY LOOKING CAREFULLY TO THE MOTORS, PRESS <ENTER>, IF ALL OF THEM WORK PERFECTLY, OR MOVE THE CURSOR TO PROBLEM AND PRESS <ENTER>.

NO_PROBLEM **PROBLEM**

ONE OF THE DRIVE MOTORS DOESN'T WORK, <PRESS ANY KEY> TO DEFINE WHICH ONE.

(The user should press any key)

MOVE THE CURSOR TO THE SUITABLE DRIVE MOTOR WHICH YOU NOTICED DOESN'T WORK AND PRESS <ENTER>.

<u>PAPER DRIVE MOTOR</u> PEN_DRIVE_MOTOR

PEN_SOLENOID CAROUSEL_STEPPER_MOTOR

<PRESS ANY KEY>, TO DEFINE THE DAMAGED COMPONENTS.

(The user should press any key)

PUT THE OSCILLOSCOPE'S PROBE AT POINT (ENCAX), OR (ENCBX). THEN ROTATE THE MOTOR MANUALLY (BY USING ONE OF THE BUTTONS ON THE FRONT PANEL OF THE PLOTTER). IF THERE IS A SQUARE SIGNAL PRESS <ENTER>, IF THERE IS NO SIGNAL:

MOVE THE CURSOR TO NO, AND PRESS <ENTER>.

<u>YES</u> NO

THE ENCODER IS OK, <PRESS ANY KEY>, TO CONTINUE TESTING THIS UNIT.

(The user should press any key)

SWITCH ON THE LOGIC ANALYZER, THEN CHOOSE YES, AND PRESS <ENTER>. OTHERWISE CHOOSE NO, AND PRESS <ENTER>.

<u>YES</u> NO

CONNECT POD (0), CHANNEL (0), TO POINT (A), AND PRESS (START) BUTTON, ON THE LOGIC ANALYZER FRONT PANEL. WAIT TILL THE ANALYZER GET DATA ON IT'S SCREEN, THEN CHOOSE YES, AND PRESS <ENTER>. OTHERWISE CHOOSE NO, AND PRESS <ENTER>.

YES NO

THE INPUT OF THE PAPER MOTOR DRIVERS IS OK, YOU SHOULD MAKE TEST TO ANOTHER POINTS, TO DEFINE THE DAMAGED COMPONENTS, <PRESS ANY KEY>, TO CONTINUE.

(The user should press any key)

CHOOSE YES, THEN PRESS <ENTER>, TO SEND THE SUITABLE SETUP. OTHERWISE CHOOSE NO, AND PRESS <ENTER>.

YES NO

CONNECT POD (0), CHANNEL (0), TO POINT (A), AND POD (0), CHANNEL (1), TO POINT (E), AFTER THAT RUN (THE INTERNAL TEST PROGRAM), AND PRESS (START) BUTTON, ON THE LOGIC ANALYZER FRONT PANEL. WAIT TILL THE ANALYZER GET DATA ON IT'S SCREEN, THEN CHOOSE YES, AND PRESS <ENTER>.

OTHERWISE CHOOSE NO, AND PRESS <ENTER>.

YES NO

THE PROBLEM IS DEFINED, THE EXPECTED DAMAGED COMPONENTS ARE:

(1) U2.

(2) Q4,Q8

(3) CR5

PRESS <ENTER>, THEN <Q>, TO EXIT.

The preceding examples are just some of the examples of fault cases with which EXP-Test System has been tested.
These tests indicate that it is reasonable to expect the EXP-Test System to be able to diagnosis better than 70% of faults in the HP 7475A Graphics plotter .
The flow chart of EXP-Test System is shown in appendix A.
Some conclusions on this research and some directions for further research will be given in the next chapter.

CHAPTER 8

CONCLUSION

Faults due to electronic system failure or malfunction are the major cause of downtime in highly automated CNC machine tool workshops. Failure detection and repair is problematic if highly skilled and well trained local maintenance staff are not in constant supply. In this dissertation it is proposed that the enhancement of the rate of repair, notwithstanding uncertainty about the skill level, is possible with an extension of the basic 'tool kit' so that expert knowledge is available in an intelligence based workstation.

Based on the conclusion that total maintenance repair knowledge may not be encapsulated a priori, the role of an expert system is rendered limited. However acknowledging that 'deep' knowledge of electronic circuitry is normally documented in graphical form, a system has been proposed and implemented that allows the user to call on graphical information (available from a CAD data-base) whilst proceeding with linguistic, diagnostic instruction from the expert system. Integrating graphical circuit knowledge and facilitating interrogation of the problem against the background of a workstation defined in this way establishes the context and reflects the normal repair situation. The normal repair situation in electronics is characterized by uncertainty, incomplete knowledge, mixed causality and in consequence a rate of convergence to fault diagnosis varying widely.

In this project the role of expert systems in electronic fault diagnosis has been reviewed, the significance of electronic systems failure in CNC machine downtime determination has been stated, a workshop facility suited to repair with maximum support has been defined, and an example workstation implemented based on an integration of hardware and software with instrument interrogation complementing instruction or direction from an expert system and circuit data reproducible on request from a CAD data-base.

Substantial additional work is required to produce a flexible and usable system that will enable the development of a workstation that will accommodate the updating necessary to cope with new repair situations. However with CAD data-base information available from machine manufacturers it is conceivable that a flexible intelligent repair station can be produced.

REFERENCES

[1] Berutti, A.: "Decentralized Maintenance for Automated Plant", Electr. Constr. and Maint. (USA), Vol. 83, No. 3, PP.: 65-69, March 1984

[2] Barbara Y.; White and John R.; Frederiksen: "Qualitative Understanding of Electrical System Troubleshooting", SIGART NEWSLETTER, Journal paper, 1985

[3] Webb, A.: "Solving the NC Maintenance Problem", MOD. MACH. SHOP (USA), Vol. 52, No. 2, PP.: 90-94, July 19

[4] Moorhead, J.: "Getting the Handle on NC. VII. Programming, Maintenance, Operation" MOD. MACH. SHOP (USA), Vol. 52, No. 6, PP.: 118-122, Nov. 1979

[5] Switzer, G.: "Hydraulic Maintenance Program Boosts NC Productivity". Tool. and Prod. (USA), Vol. 43, No. 1, PP.:60-62, April 1977

[6] Kolenaty, G.; Holland, M.: "SYS/PLANR: A Decision-Support System for Managing Software Development", Softfair. A conference on software development tools, Techniques, and Alternative, Proceedings PP.: 99-108, 25-28 July 1983

[7] Thrasher, D.; Giomi, M.: "Drill Machine Maintenance Service", Circuits Manuf. (USA), Vol. 22, No. 6, PP.:58-59, June 1982

[8] Davis, W.J.; Carnahan, J.V.: "Decision Support for Road Surface maintenance", Dmega (GB), Vol. 15, No. 4, PP.: 313-322, 1987

[9] Stuart C. Shapiro; Sargur N. Srihari; Ming-Ruey Taie; James Geller: "Development of an Intelligent Maintenance Assistant", SIGART Newsletter, Vol. 92, PP.:48-49, April 1985

[10] Eva Hudlika and Victor R.: "Construction and use of a Causal Model of a problem-Soling System", SIGART Newsletter (acm), Vol. 93, July 1985

[11] Kothari, D.P.; Girotra, S.N.: "New Optimal and Reliable Generator Maintenance Scheduling Algorithm", J. Inst. Eng. (India) Electr. Eng. Div. (India), Vol. 67, pt.EL4, PP.:171-173, Feb. 1987

[12] Escudero, L.F.: "On Maintenance Scheduling of Production Units", Eur. J. Oper. Res. (Netherlands), Vol. 9, No. 3, March 1982

[13] Kathari, D. P.; Girotra, S.N.; Yadava, G.S.: "Some Aspects of Optimal Maintenance Scheduling of Generating Units", J. Inst. Eng. (India) Electr. Eng. Div., Vol. 66, pt.EL1, PP.:41-45, Aug. 1985

[14] Konstantionova, I.M.; Raikin, A.L.: "Optimal Scheduling of Operation Involving Temporary Equipment Stoppage in Continuous Process Systems", Avtom. and Telemekh.(USSR), Vol. 42, No. 6, PP.: 170-180, June 1981

[15] Vergin, R.C.; Scriabin, M.: "Maintenance Scheduling for Multicomponent Equipment", AIIE Trans. (USA) Vol. 9, No. 3, PP.: 297-305, Sept. 1977

[16] Park, K.S.: "Optimal Scheduling of Multiple Preventive Maintenance Activities", Microelectron and Reliab. (GB), Vol. 23, No. 2, PP.: 351-354, July 1983

[17] Yamayee, Z.; Sidenblad, K.; Yoshimura, M.: "A Computationally Efficient Optimal Maintenance Scheduling Method", IEEE Trans. Power Appar. and Syst. (USA), Vol. PAS-102, No. 2, PP.: 330-338, Feb. 1983

[18] Sivazlian, B. D.: "Optimum Scheduling of a New Maintenance Program Under Stochastic Degradation", Microelectron Reliab. (UK), Vol. 29, No. 1, PP.: 57-71, 1989

[19] Smiley, R. G.: "Rotation Machinary: Monitoring and Fault Diagnosis", Sound and Vib. (USA), Vol. 17, No. 9, PP.:26-28, Sept 1983

[20] Roger T. Hartly: "CRIB: Computer Fault-finding Through Knowledge Engineering", Computer, March 1984

[21] Sylla, C.: "Diagnosis Interval Estimation in Process Analysis for Troubleshooting Tasks", Computer, Ind. Eng. (UK), Vol. 17, PP.: 519-524, 1989

[22] Neun, J.A.; Walter, T.J.; Martin, C.H.; Pyrce, D.J.: "Development of Techniques to Diagnose the Health of Metal Cutting Machinary", Sensors, 85 Conference, PP.: MS85-1000/1-12, Nov. 1985

[23] Rault, A.; Brunet, J.; Deviq, A.; Ephraim, P.: "Modelling and Identification for Machine Tool Fault Detection and Diagnosis", Proceeding of the 25th IEEE Conference on Decision and Control (Cat. No. 86 CH 2344-0), PP.:808-12, Vol. 2, Dec. 1986

[24] James S. Bennett: "DART: An Expert System for Computer Fault Diagnosis", Augast 1981

[25] Randall Davis; Howard Shrobe; Walter Hamscher: "Diagnosis Based on Description of Structure and Function", AAAI 1982, Vol. 82, PP.: 137-142, 1982

[26] R. Davis; H. Shrobe: "The Hardware Troubleshooting Group", SIGART Newsletter, Vol. 93, PP.: 17-20, July 1985

[27] Robert Neches; William R. Swartout; Johanna Moore: "Explainable (and Maintenable) Expert Systems", IJCAI-85, PP.:

382-389, August 1985 Los Angles

[28] Beichelt, F.; Fischer, K.: "Optimum Control of Automatic Systems Under Uncertainty", Control Science and Technology for the Progress of society. Proceedings of the Eighth Triennial World Congress Of the International Federation of Automatic control, PP.: 875-880, Vol. 2, Aug. 1981

[29] A. R. Dormer: "Real Time Process Optimisation", Factory 2000 Integrating Information and Material Flow (Publ. No. 80), 31 Aug- 2 Sep 1988

[30] Pattipati, K. R.; Shaw, J. J.; Deckert, J. C.; Beean, L. K.; Alexandridis, M.G.; Lougee, W.P.: "Confidante: A Computer Based Design Aid for the Optimal Synthesis, analysis and operation of maintenance facilities", Proceedings of Autotest Con. 84 (Cat. No. 84 CH1921-6), PP.: 390-404, Nov. 1984

[31] Smirnov, A. N.: "Experience in Organizing the Servicing of NC Machine Tools", Sov. Eng. Res. (GB), Vol. 2, No. 2, Feb. 1982

[32] Mikhailov, O. p.; Veselov, O. V.: "Promising Developments in Microprocessor Drives (for Machine Tools)", Sov. Eng. Res. (GB), Vol. 3, No. 12, Dec. 1983

[33] Chapurlat, A.; de Bonnieres, P.; Kretzschmar, J.; Billiau, R.(Editors): "Moise, an Expert System for NC Machine-Tool Trouble Diagnosis", Proceedings of Operative Expert Syetems. International Seminar-Exhibition, April 1987

[34] Keller, A. Z.; Kamath, A. R.R.; Perera, U.D.: "Reliability Analysis of CNC Machine Tools", Reliab. Eng. (GB) Vol. 3, No. 6, Nov 1982

[35] Padraig Cunningham; Michael Brady: "Knowledge Based Systems for Fault Diagnosis", AICS, Irish Conference on 'AI' and

Cognitive Science, UCD Dublin, Sep. 1988

[36] Brown J.S.; Bruton R.R.; deKleer J.: "Pedagocial, Natural Language and Knowledge Engineering Techniques in SOPHIE I,II and III", Intelligent tutoring system, eds. D.Sleeman, J.s.Brown, PP.: 227-282, Academic Press, 1982

[37] Clancey W.J.: "The Epistemology of a Rule-Based Expert System - a Framework for explanation", Artificial Intelligence, Vol. 20, PP.: 215-251, 1983

[38] Cunningham P. et al.: "Diagnostic Heuristics and Perspectives", Expert Systems, PP.: 242-253, 1986

[39] Padraig Cunningham: "A Taxonomy of Knowledge Representation Formalisms", Trinity College Dublin Computer Science Report No. CSC-87-01, 1987

[40] Dague P.; Raiman O.; Deves P.: "Troubleshooting: When Modeling is the Trouble", AAAI, 1987

[41] Davis R.: "Diagnostic Reasoning Based on Structure and Behavior", Artificial Intelligence, Vol. 24, PP.: 347-410,1984

[42] deKleer J.: "How Circuits Work", Artificial Intelligence, Vol. 24, PP.: 205-280, 1984

[43] Feigenbaum E.A.: "The Art of Artificial Intelligence: Themes and Case Studies of Knowledge Engineering", IJCAI-77, PP.: 1014-1029, 1977

[44] Genesereth M.R.: "The Use of Design Descriptions in Automated Diagnosis", Artificial Intelligence, Vol. 24, PP.: 411-436, 1984

[45] Giovannini F.; Malabocchia F.: "A Model Based Expert System for HW Troubleshooting Driven by Compiled Control Knowledge", Expert Systems, PP.:65-77, 1985

[46] Pople H.E.; Jr.; Myers J.D.; Miller R.A.: "DIALOG: A Model of

Diagnostic Logic for Internel Midicine", IJCAI, PP.: 848-855, 1975

[47] Shortliffe E.H.: Computer-Based Medical Consultation: MYCIN, American Elsevier, 1976

[48] Weiss S.M.; Kulikowski C.A.; Safir A.: "A Model Based Consultation System for the Long-Term Managment of Glaucoma", IJCAI, PP.:826-832, 1977

[49] Padraig Cunningham; Michael Brady: "Qualitative Reasoning in Electronic Fault Diagnosis",IJCAI 1987, PP.: 443-445,

[50] G.M. O'hare; D.A.Bell: "The Coexistence Approach to Knowledge Representation", Expert Systems, Vol. 2, No. 4, Oct. 1985

[51] A.S. Pollitt: "Reducing Complexity by Rejecting the Consultation Model as a Basis for the Design of Expert Systems", Expert Systems, Vol. 3, No. 4, Oct. 1986

[52] Michael D. Rychener: "Expert Systems for Engineering Design", Expert Systems, Vol. 2, No. 1, Jan 1985

[53] M. Hofmann; J. Caviedes; J. Bourne; G. Beale; A. Brodersen: "Building Expert Systems for Repair Domains", Expert Systems, Vol. 3, No. 1, Jan 1986

[54] Michael R. Genesereth: "Diagnosis Using Hierarchical Design Models", AAAI, PP.: 278-283, Aug. 1982

[55] Johan de Kleer; Brian C. Williams: "Reasoning About Multiple Faults", AAAI, PP.:132-139, 1986

[56] Hal Shubin; John Wade Ulrich: "IDT: An Intelligent Diagnostic Tool", AAAI, PP.:290-295, Aug 1982

[57] Ramesh S. Patil; Peter Szolovits; William B. Schwartz: "Causal Understanding of Patient Illness in Medical Diagnosis", IJCAI, PP.: 893-899, Aug 1981

[58] T.R. Addis: "Towards an 'Expert' Diagnostic System", ICL

Technical Journal, Vol. 2, May 1980

[59] Sanjay Mittal; B. Chandrasekaran; Jon Sticklen: "Patrec: A Knowledge-Directed Database for a Diagnostic Expert System", Computer, Sep. 1984

[60] L.F. Pau: "Survey of Expert Systems for Fault Detection, Test Generation and Maintenance", Expert Systems, Vol. 3, No. 2, April 1986

[61] Dana S. Nau: "Expert Computer Systems", Computer, Feb. 1983

[62] Randall Davis; Howard Shrobe: "Representing Structure and Behaviour of Digital Hardware", Computer, Oct. 1983

[63] Padraig Cunningham: Knowledge representation in Electronic Fault Diagnosis, P.HD. Thesis, Trinity College Dublin, 1989

[64] Gleeson J.F.J.: An EMYCIN Based Expert System for Electronic Fault Diagnosis, M.Sc. Thesis, Dept. of Computer Science, Trinity College Dublin,1986

[65] Wally Wang; John Mueller: Illustrated VP-Expert, Wordware Pub., Inc., 1989

[66] Paperback Software: VP-Expert, 1987

[67] J. L. Alty; M. J. Coombs: Expert Systems Concepts and Examples, The National Computing Centre Limited, 1984

[68] Louis E. Frenzel, Jr.: Crash Course in Artificial Intelligence and Expert Systems, (USA) 1987

[69] John D. 1ENK: Handbook of Practical Microcomputer Troubleshooting, Reston Pub. Company, Inc., 1979

[70] Allen Buckroyd: Computer Integrated Testing, BSP (UK) 1989

[71] S.J. Cahill: Designing Microprocessor-Based Digital Circuitry, PHI 1985

[72] Paul C. Barr; Ronald L. Krimper; Michael R. Lazear; Christopher Stammen: CAD Principles and Applications, T& W

Systeems, Inc 1985

[73] Patrick R. Carberry: CAD/CAM with Personal Computers, TAB Book Inc. 1985

[74] Joe Rooney; Philip Steadman: Principles of Computer-Aided Design, Open University 1987

[75] Spyros G. Tzafestas: Knowledge-Based System Diagnosis, Supervision, and Control, Plenum Press, New York 1989

[76] Barry B. Brey: Microprocessor/Hardware Interfacing and Applications, Bell & Howell Company, 1984

[77] Keravnou E.T.; Johnson L.: "Competent Expert Systems", Kogan Page, 1986

[78] F. Hayse-Roth; D.A. Waterman; D.B. Lenat: "Building Expert Systems", Addison-Wesly, Reading, MA, 1983

[79] B.G. Buchanan; E.H. Shortliffe: "Rule-Based Expert Systems", Addison-Wesly Publishing Co., Inc., 1984

[80] M.R. Genesereth: "An Overview of MRS for AI Experts", Standford Heuristic Programming project, Memo HPP-82-27, Dept. of Computer Science, 1982

[81] C.L. Forgy: "OPS5 User's Manual", Carnegie-Mellon University, 1981

[82] Lotfi A. Zadeh: "Fuzzy Logic", Computer, PP.: 83-92, April 1988

APPENDIX (A)

SEARCH PROGRAM

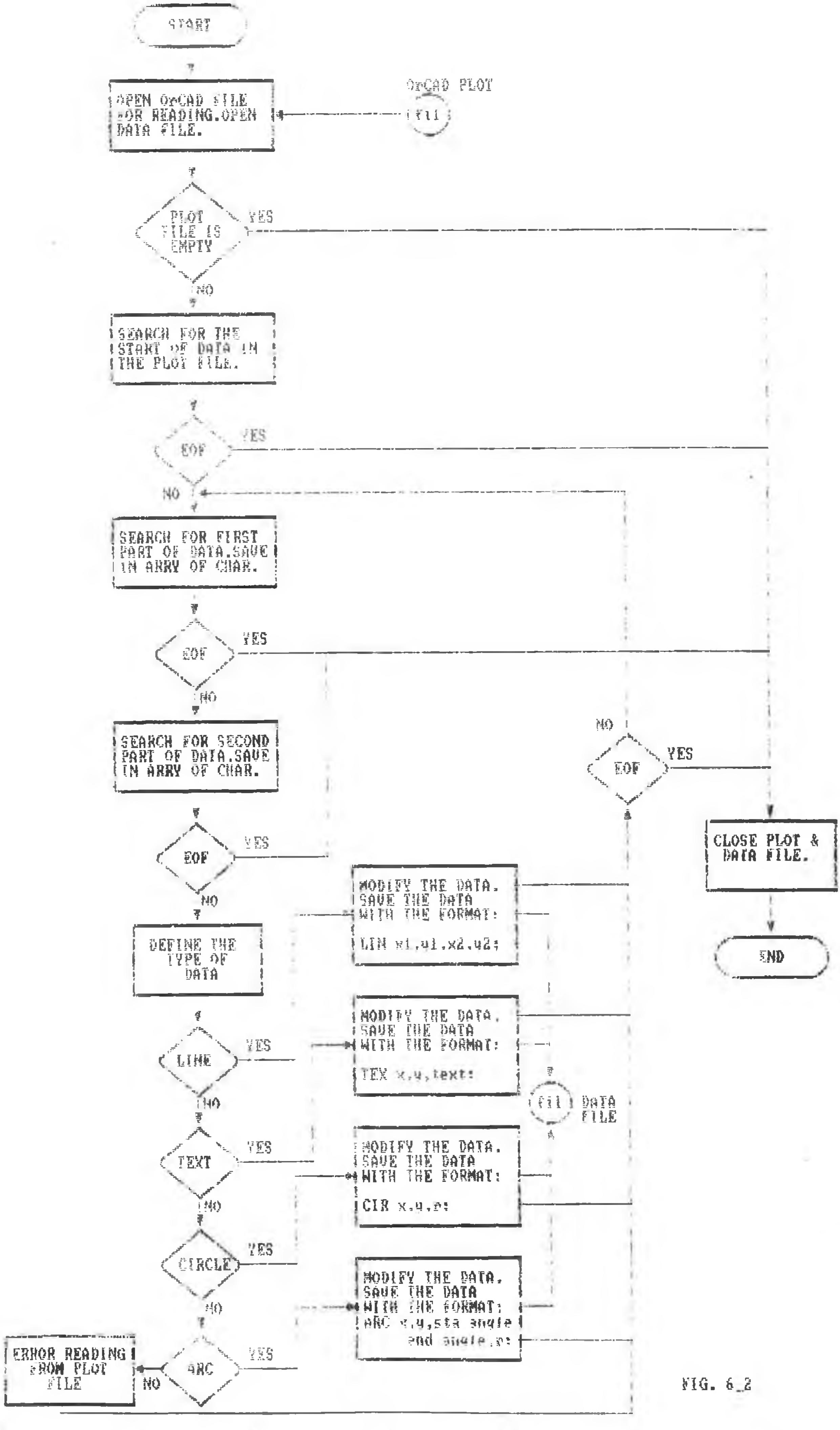
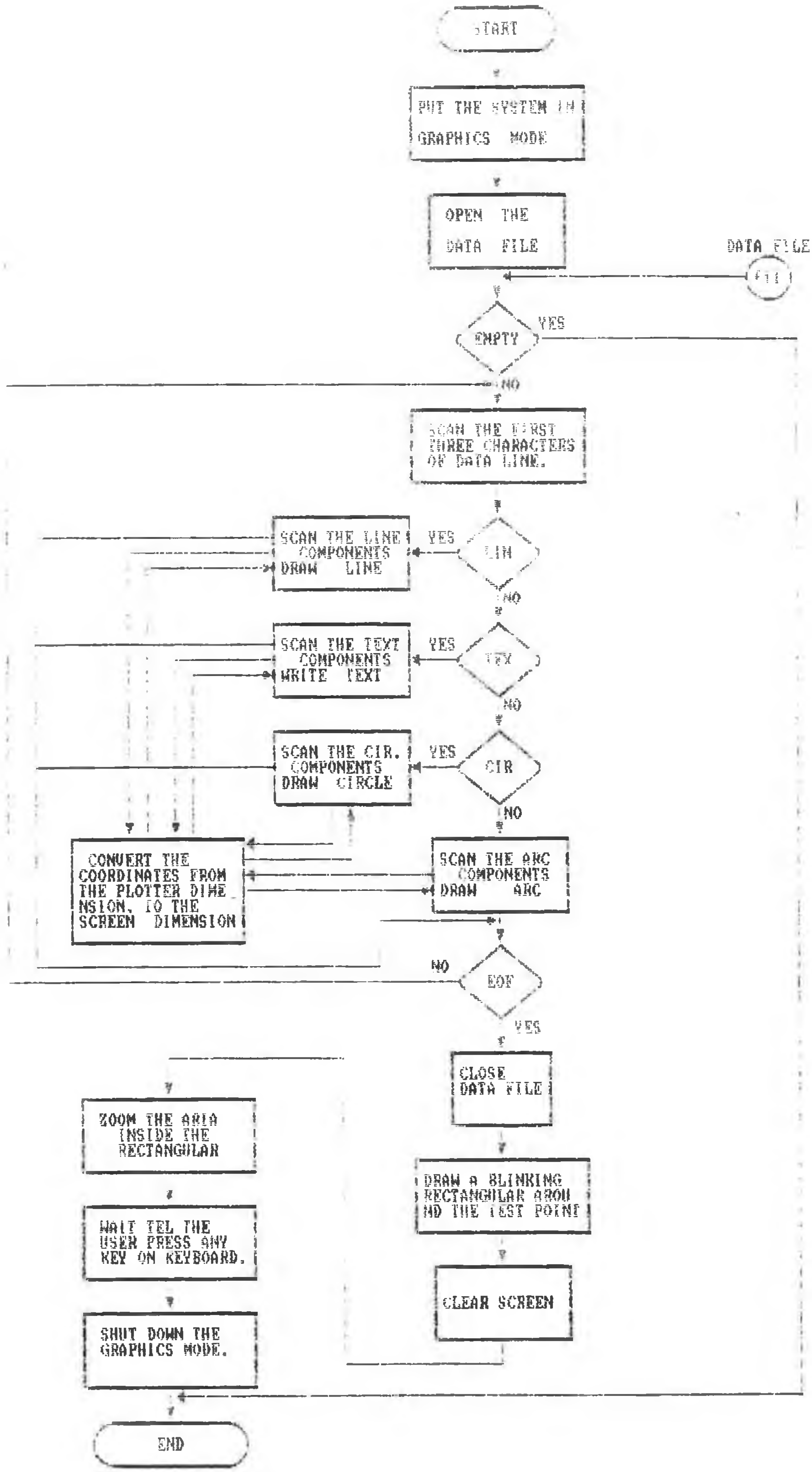
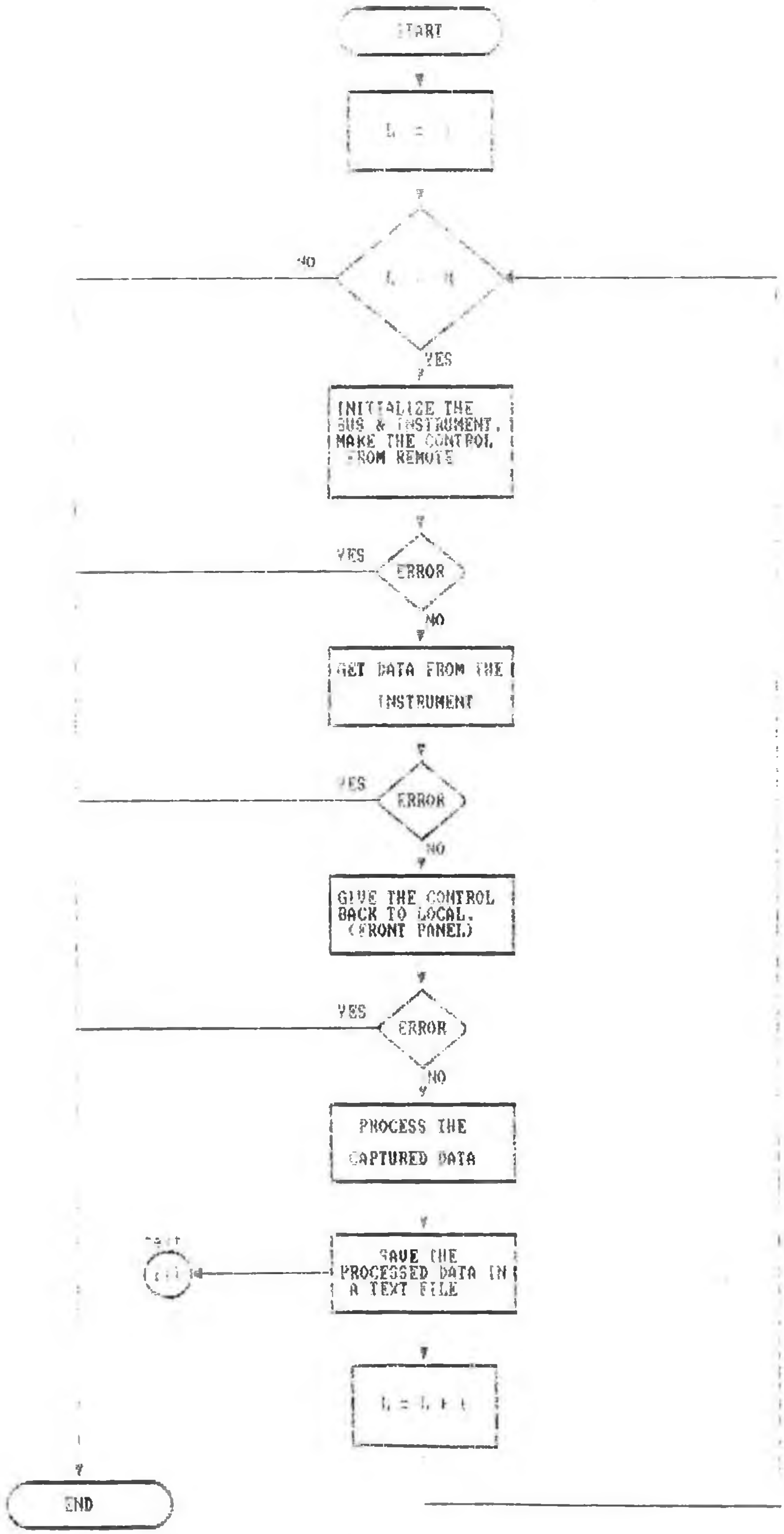


FIG. 6_2

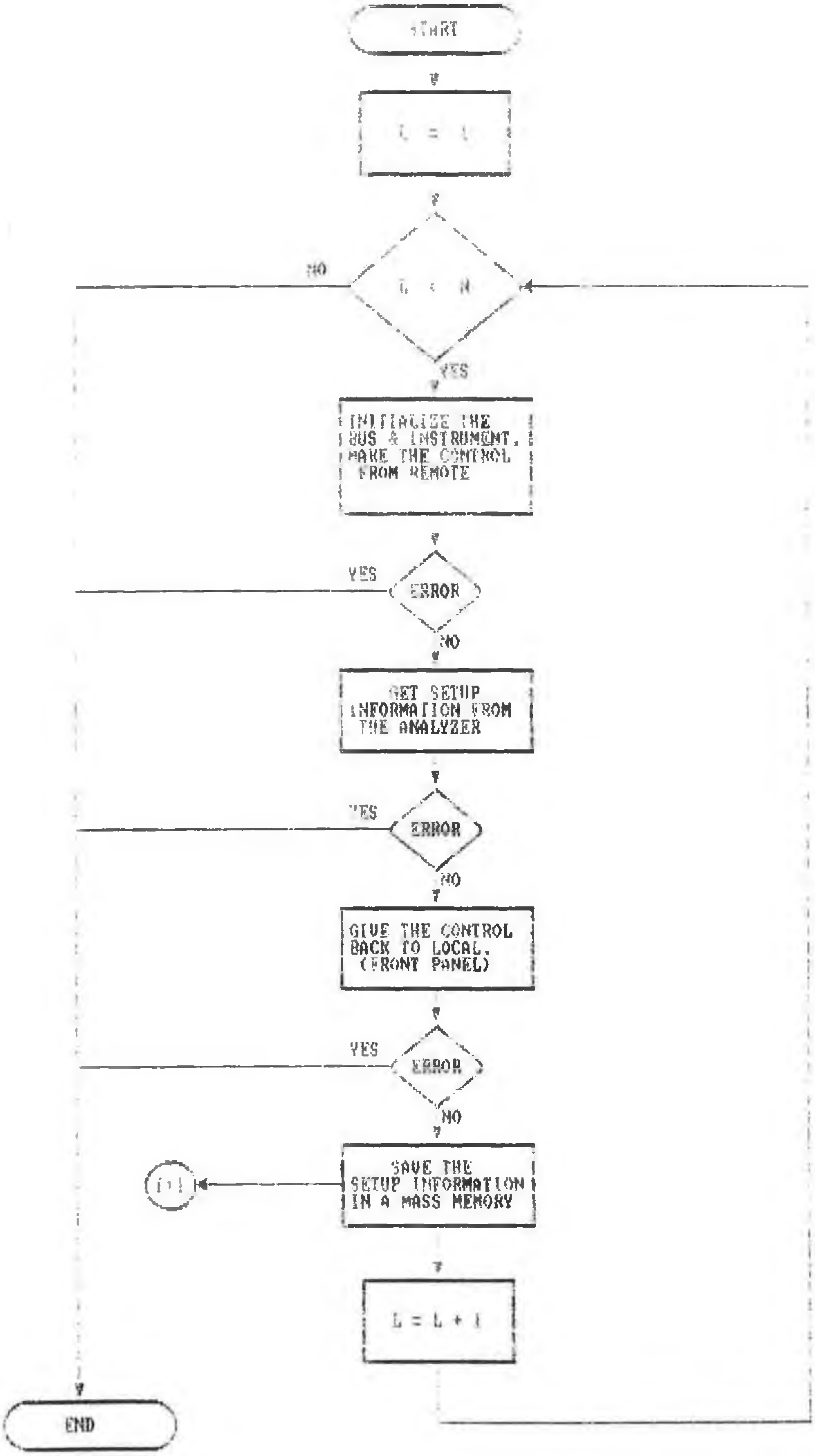
SKETCH THE SCHEMATIC DIAGRAM



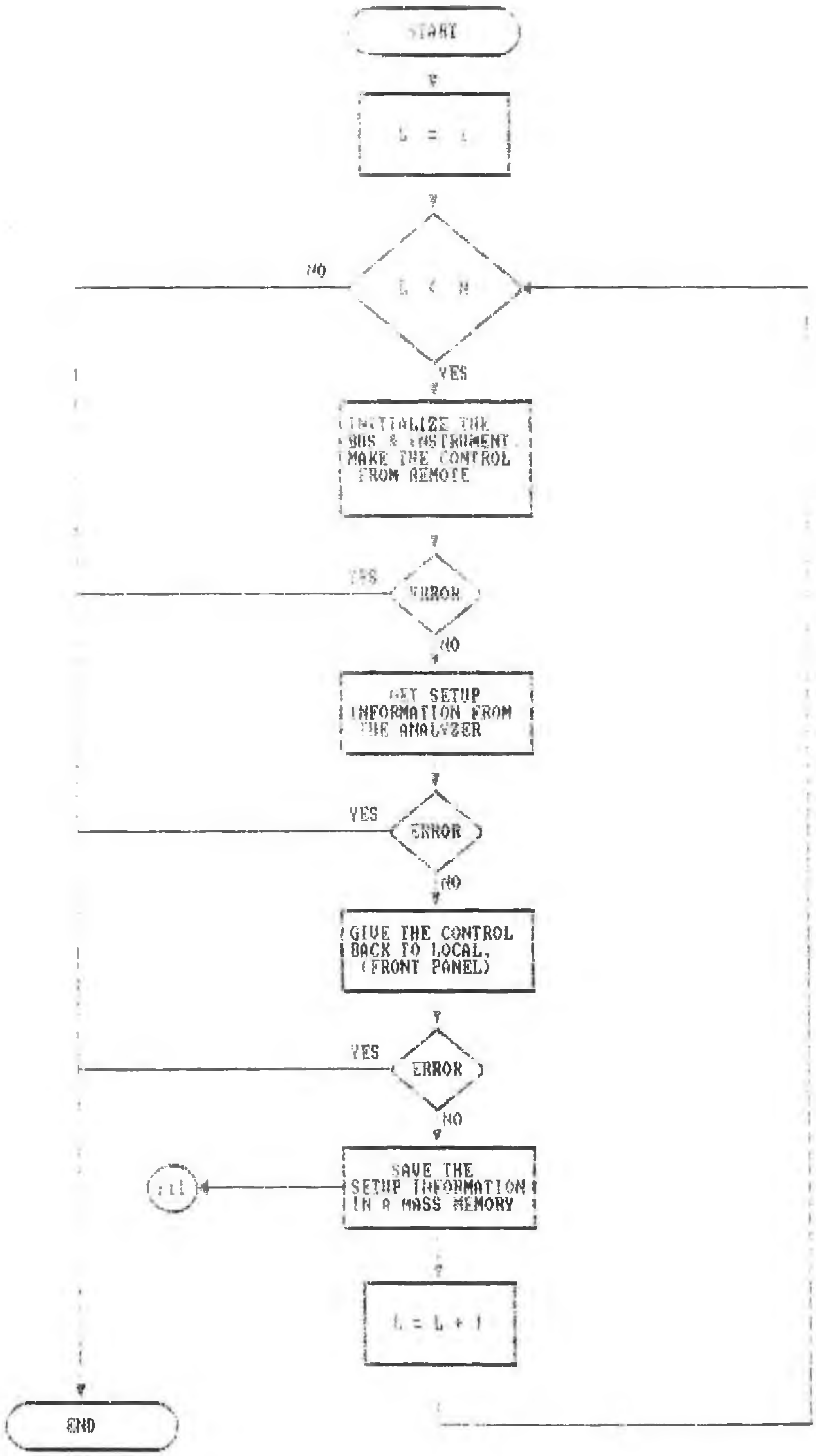
GETTING THE REFERENCE SIGNALS FROM THE OSCILLOSCOPE & THE ANALYZER



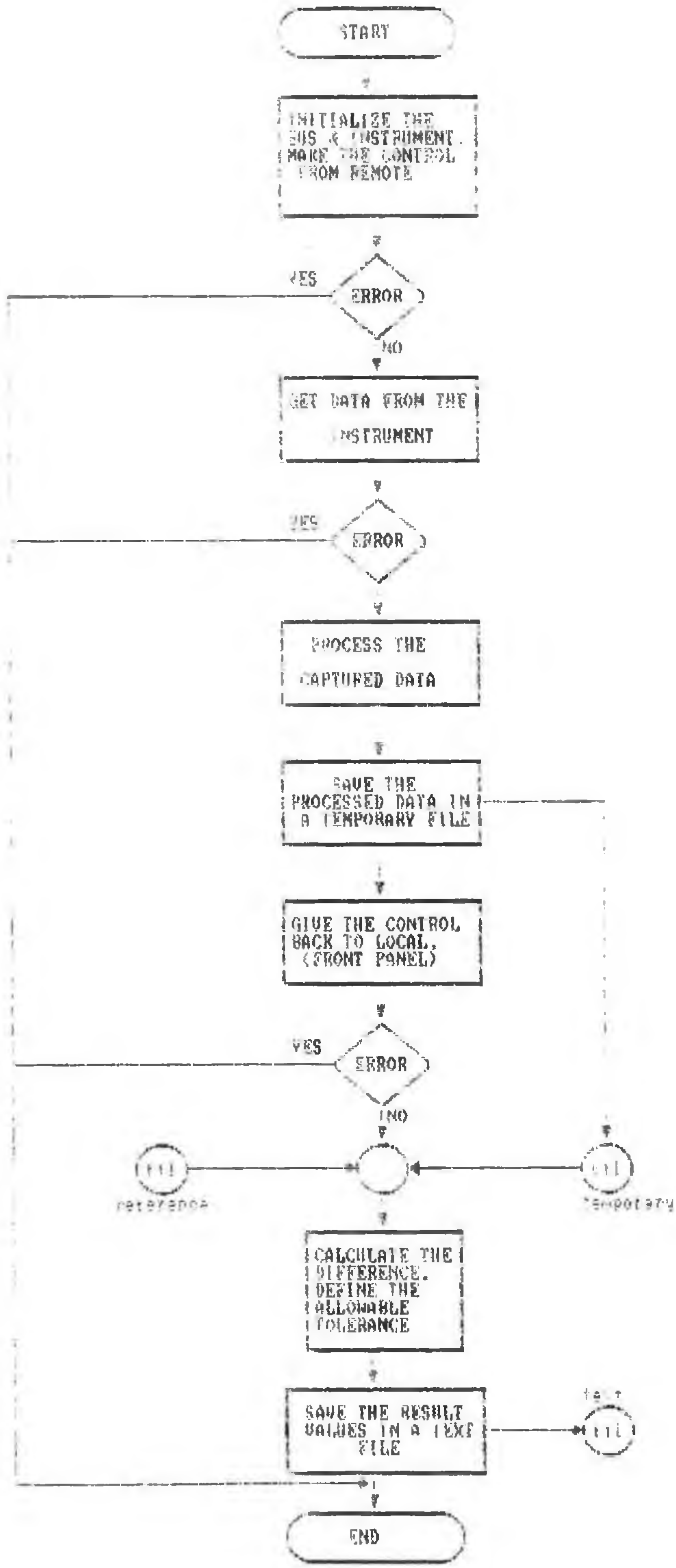
GETTING THE SETUP INFORMATION
FROM THE LOGIC ANALYZER



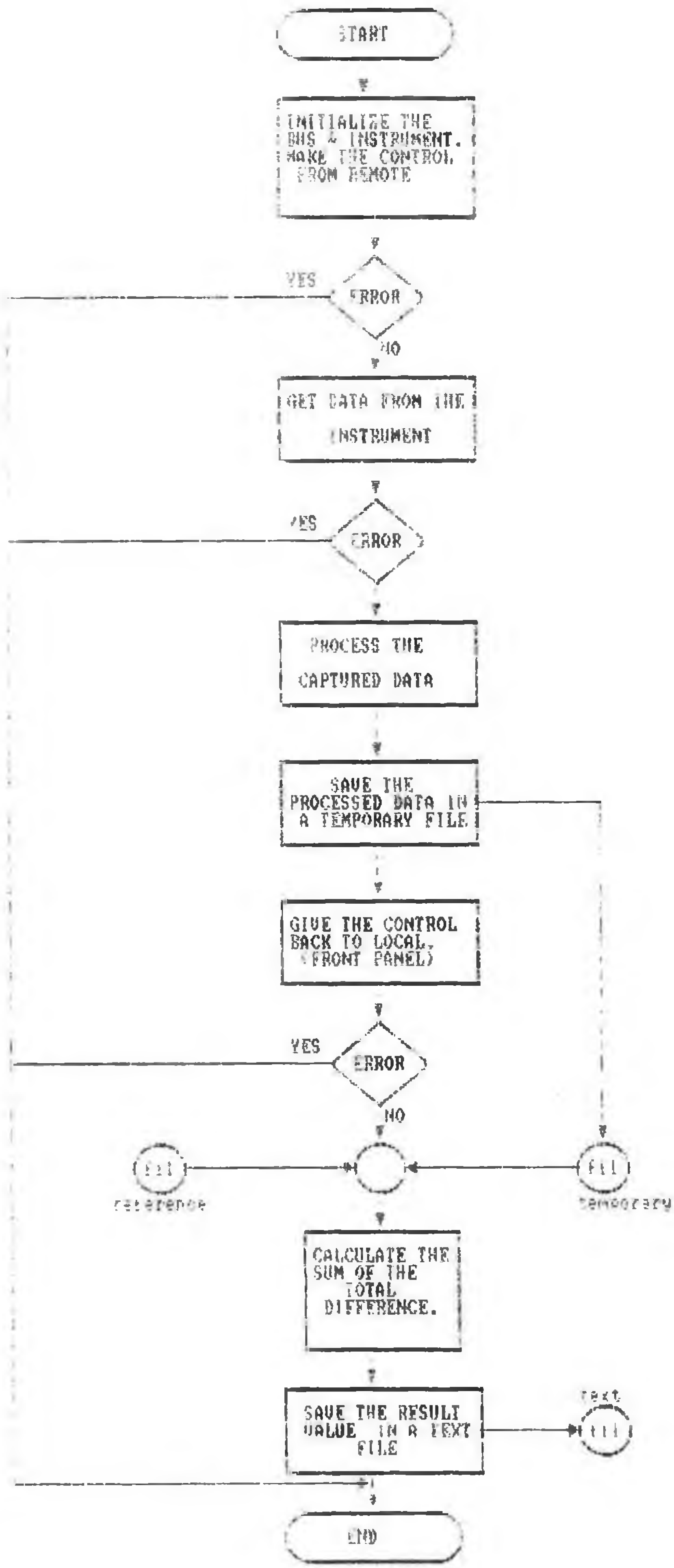
GETTING THE SETUP INFORMATION FROM THE LOGIC ANALYZER



GETTING AND COMPARING THE OSCILLOSCOPE SIGNAL

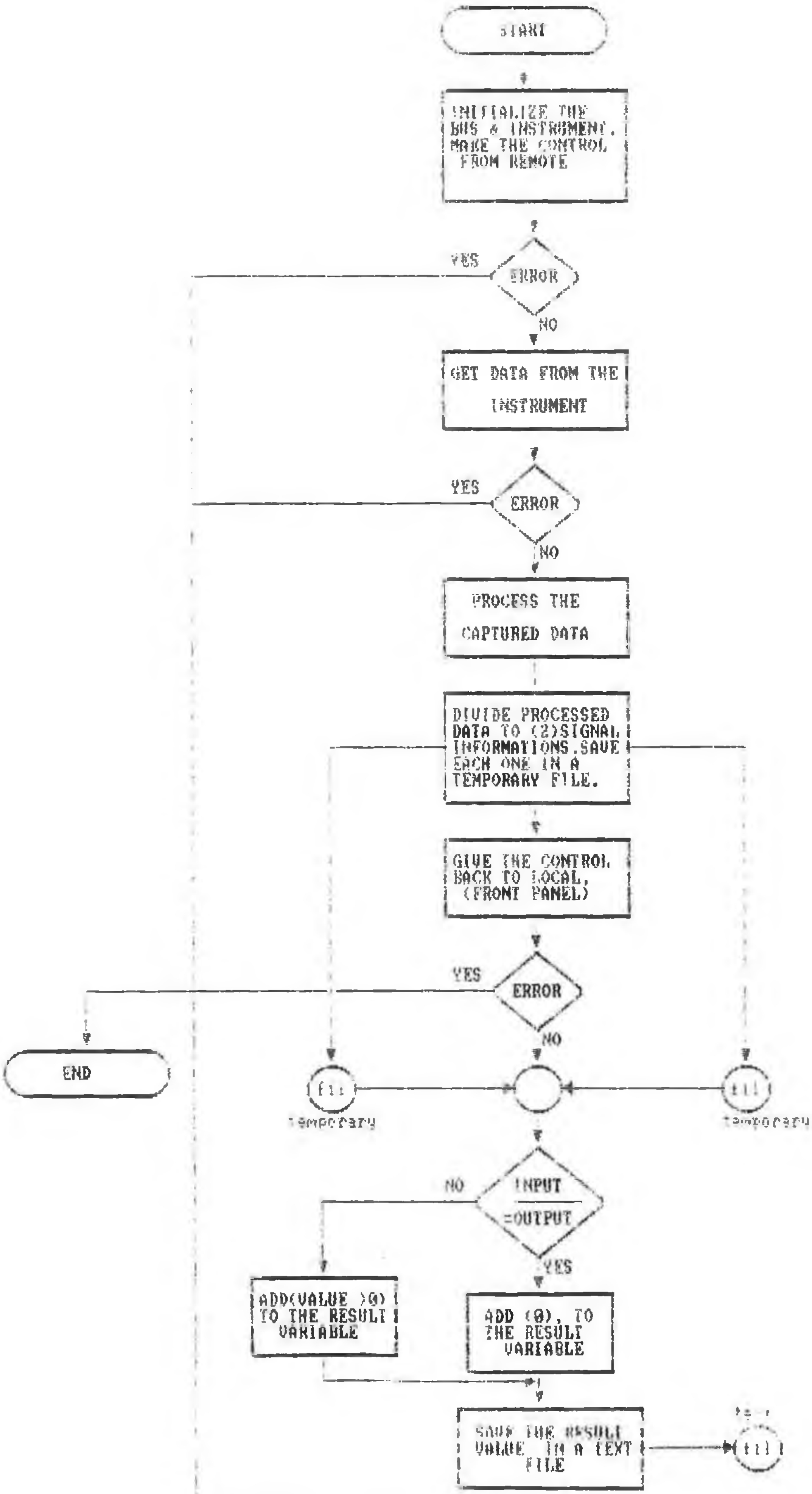


GETTING AND COMPARING THE LOGIC ANALYZER SIGNAL

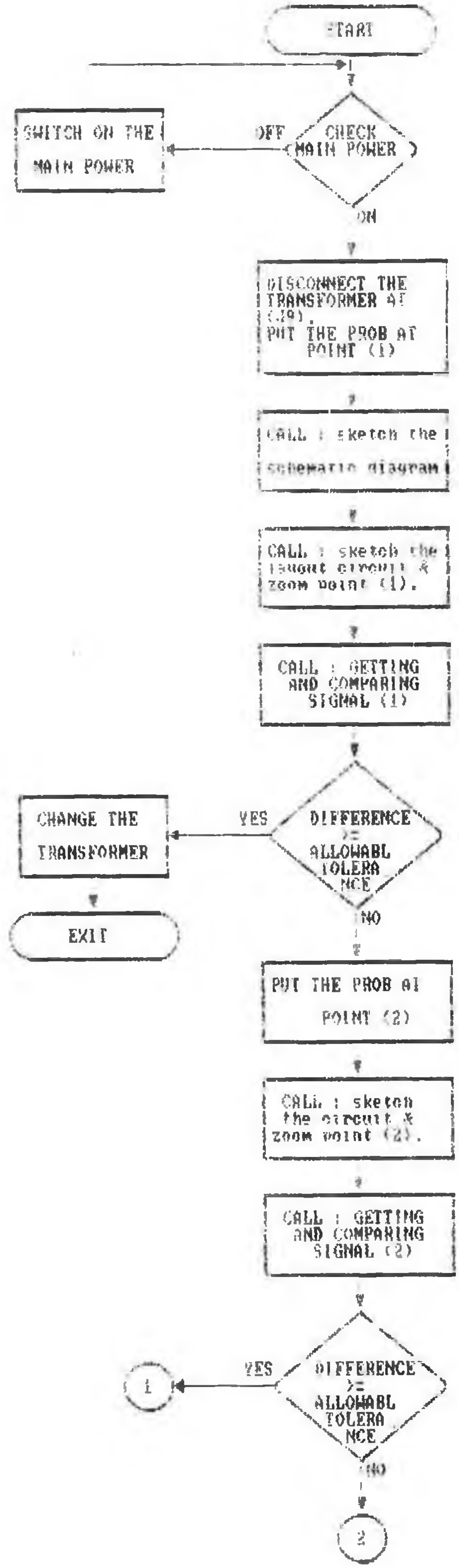


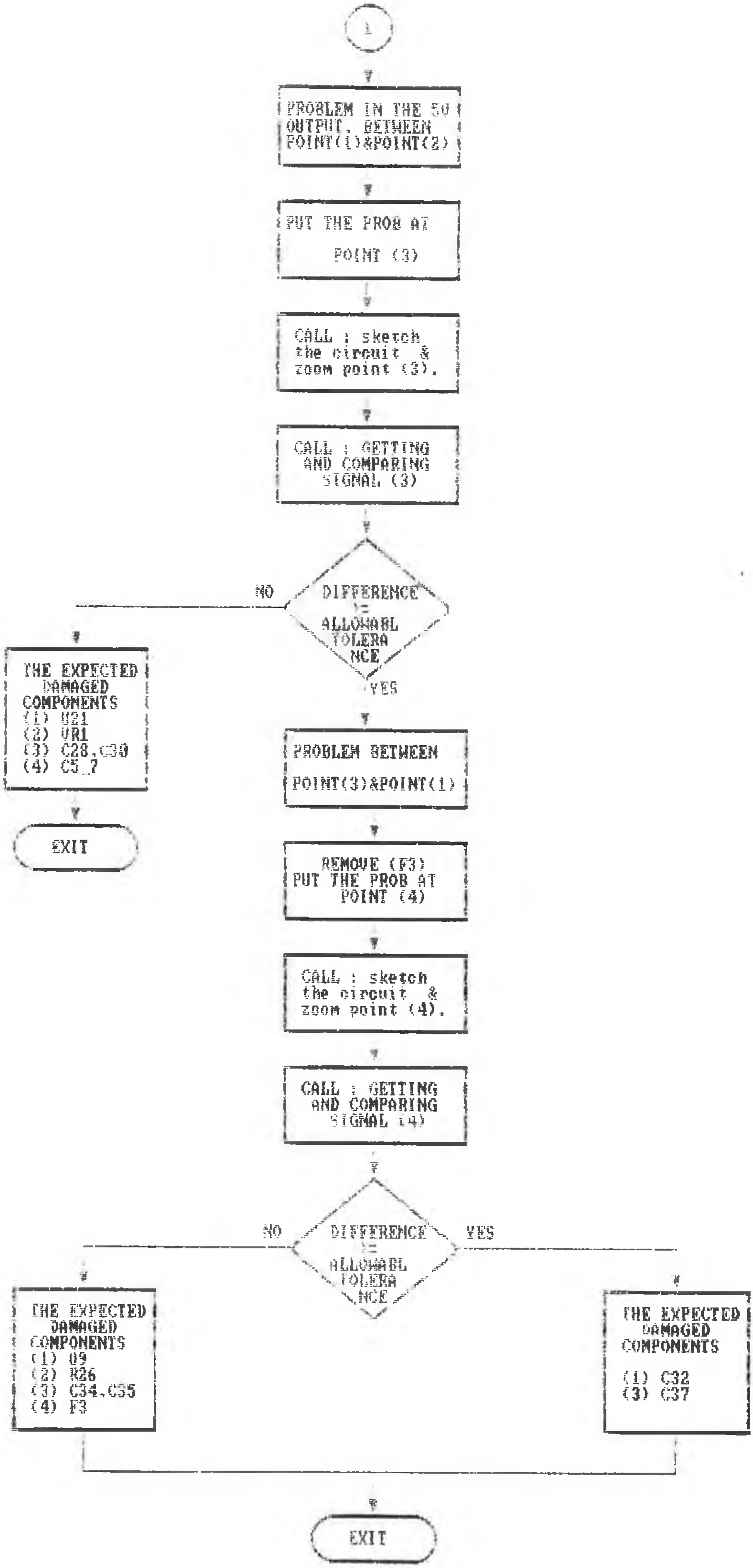
GETTING SIGNALS FROM THE LOGIC ANALYZ

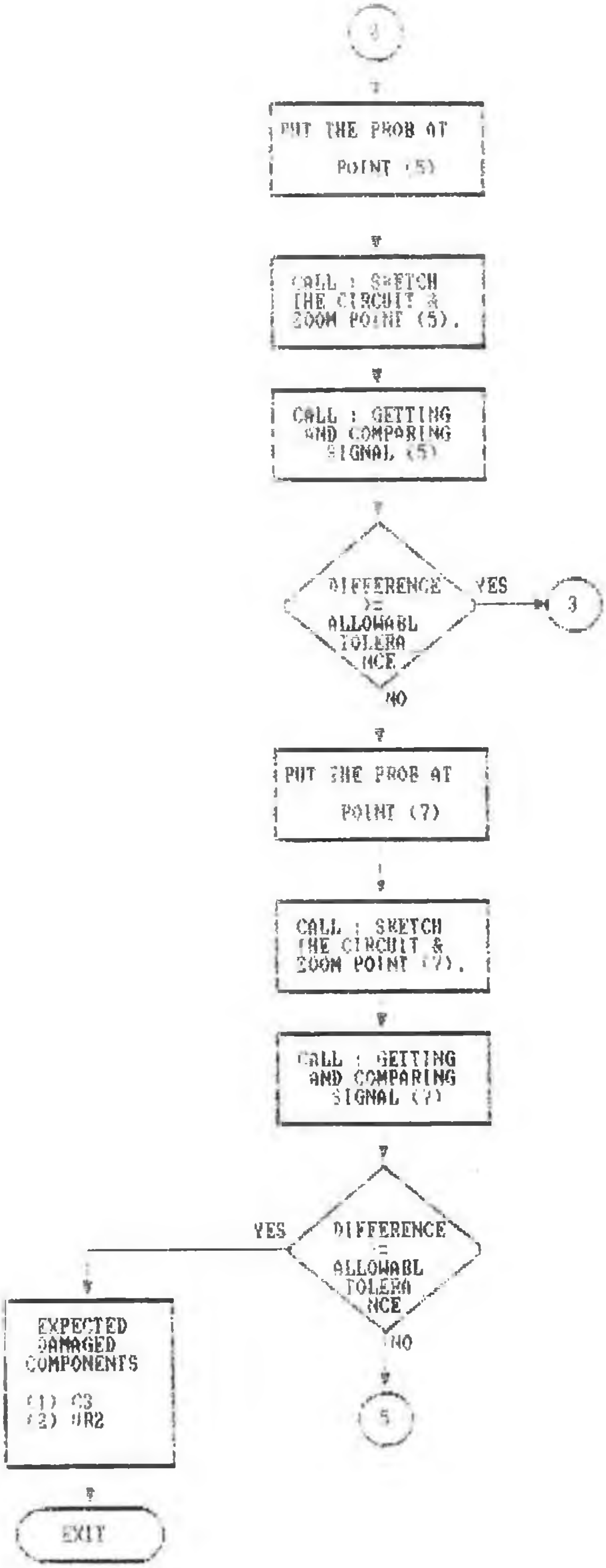
AND PROVING THAT: INPUT = OUTPUT



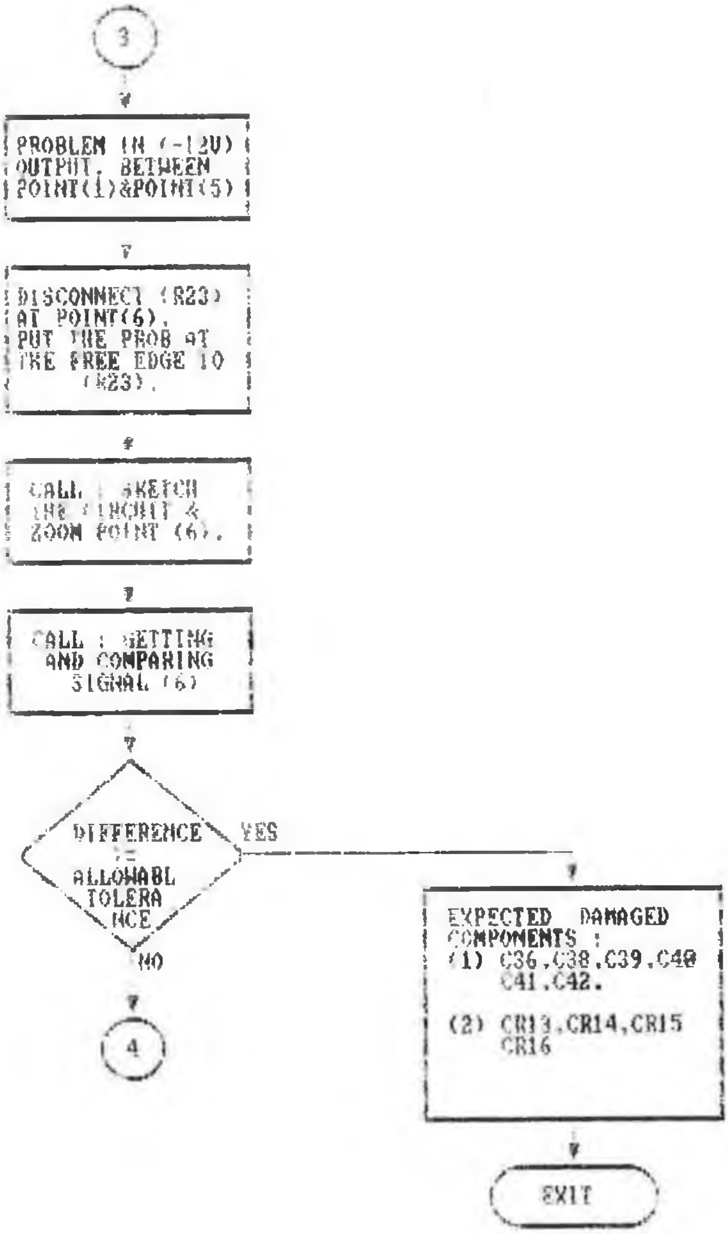
EXP_TEST SYSTEM FLOWCHART



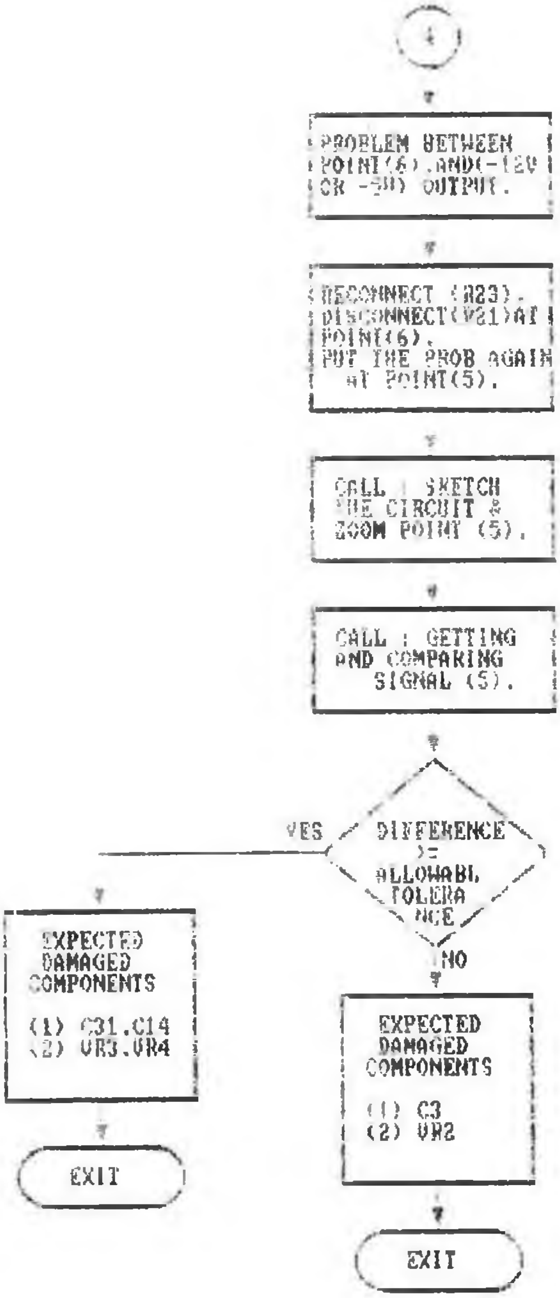

1
PROBLEM IN THE 50 OUTPUT, BETWEEN POINT(1)&POINT(2)
PUT THE PROB AT POINT (3)
CALL : sketch the circuit & zoom point (3).
CALL : GETTING AND COMPARING SIGNAL (3)
DIFFERENCE >= ALLOWABL TOLERA NCE
NO
YES
THE EXPECTED DAMAGED COMPONENTS
(1) U21
(2) VR1
(3) C28,C30
(4) C5_7
EXIT
PROBLEM BETWEEN POINT(3)&POINT(1)
REMOVE (F3) PUT THE PROB AT POINT (4)
CALL : sketch the circuit & zoom point (4).
CALL : GETTING AND COMPARING SIGNAL (4)
DIFFERENCE >= ALLOWABL TOLERA NCE
NO
YES
THE EXPECTED DAMAGED COMPONENTS
(1) U9
(2) R26
(3) C34,C35
(4) F3
THE EXPECTED DAMAGED COMPONENTS
(1) C32
(3) C37
EXIT

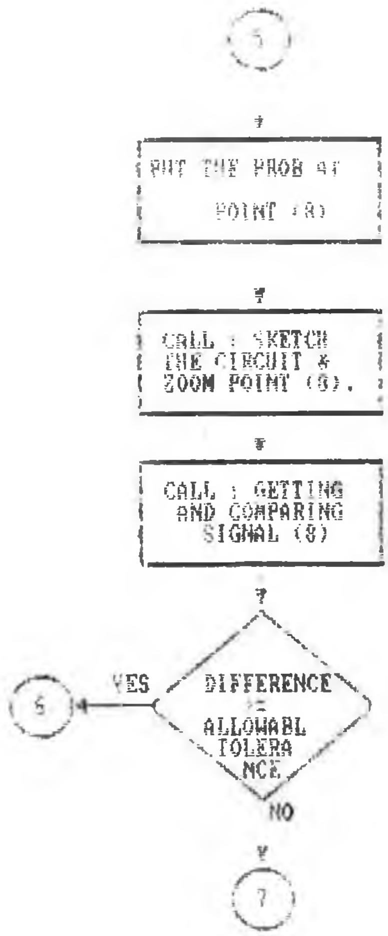


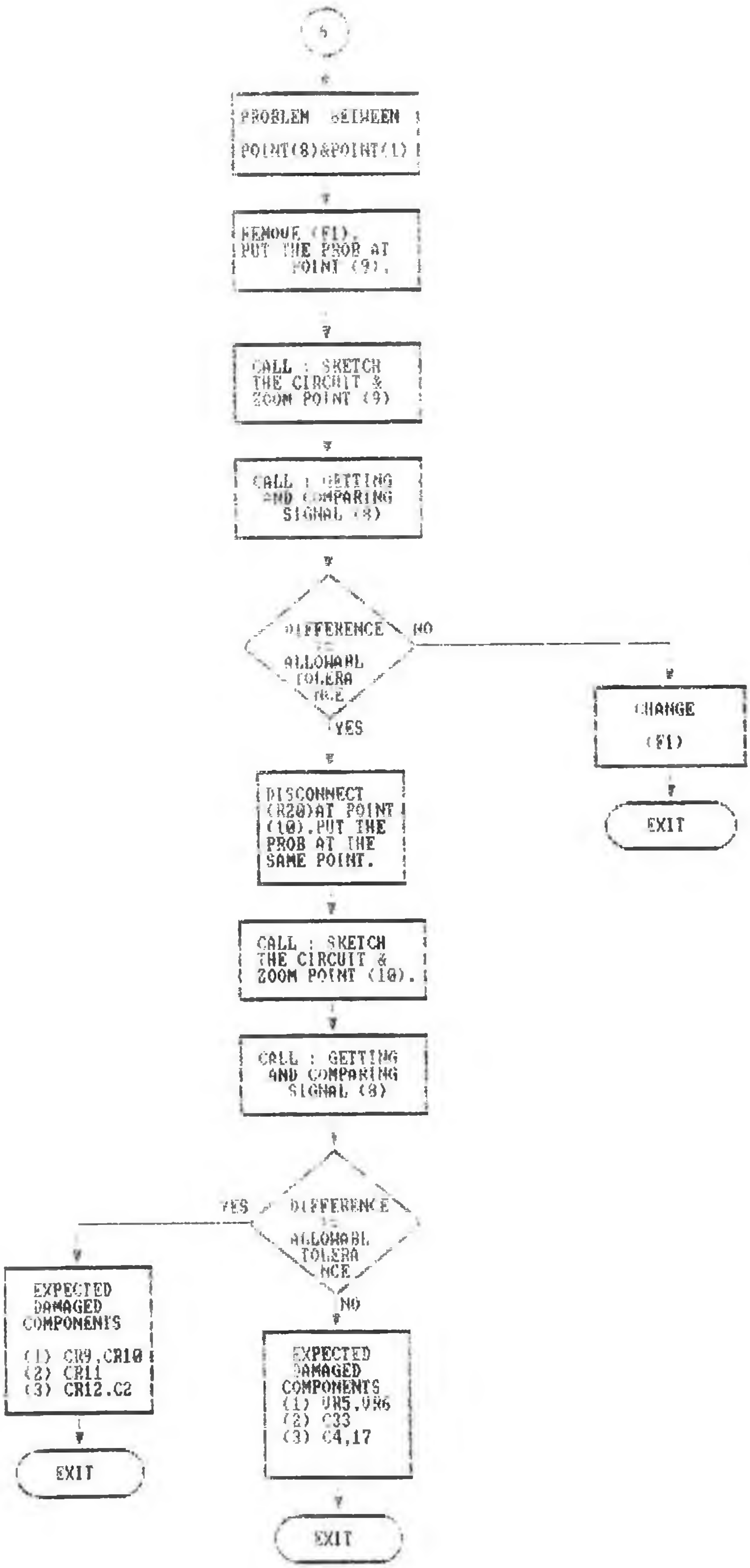
PUT THE PROB AT POINT (5)
CALL : SKETCH THE CIRCUIT & ZOOM POINT (5).
CALL : GETTING AND COMPARING SIGNAL (5)
DIFFERENCE >= ALLOWABL TOLERA NCE
YES
3
NO
PUT THE PROB AT POINT (7)
CALL : SKETCH THE CIRCUIT & ZOOM POINT (7).
CALL : GETTING AND COMPARING SIGNAL (7)
DIFFERENCE >= ALLOWABL TOLERA NCE
YES
NO
5
EXPECTED DAMAGED COMPONENTS
(1) C3
(2) VR2
EXIT

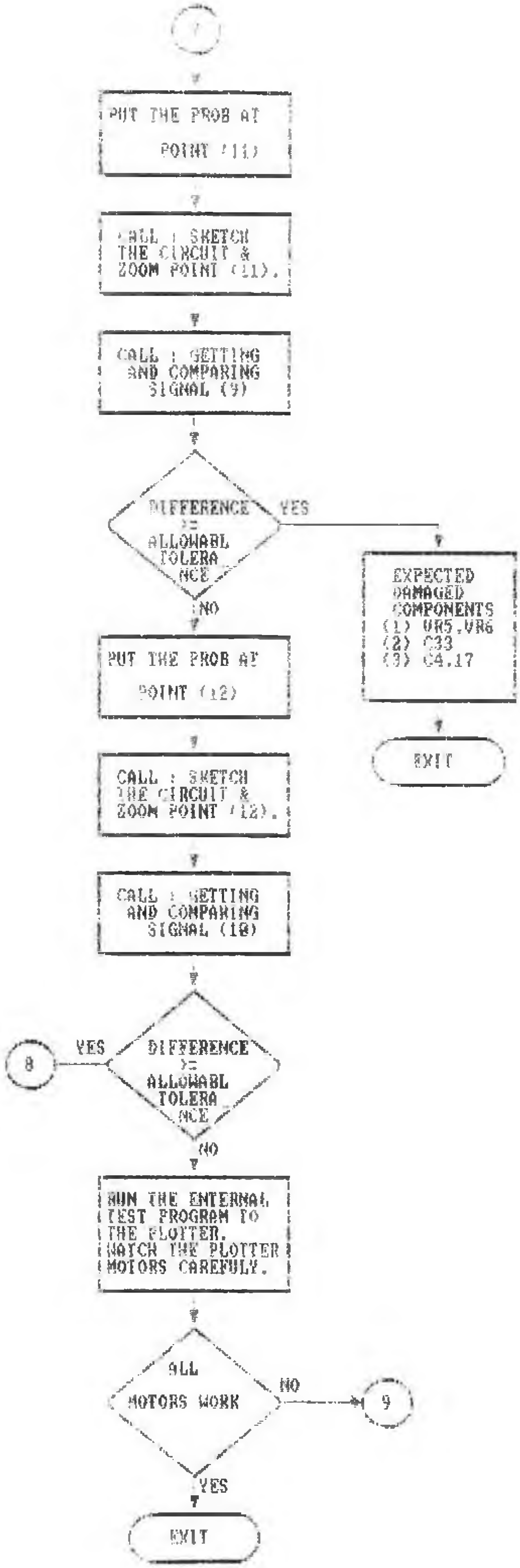


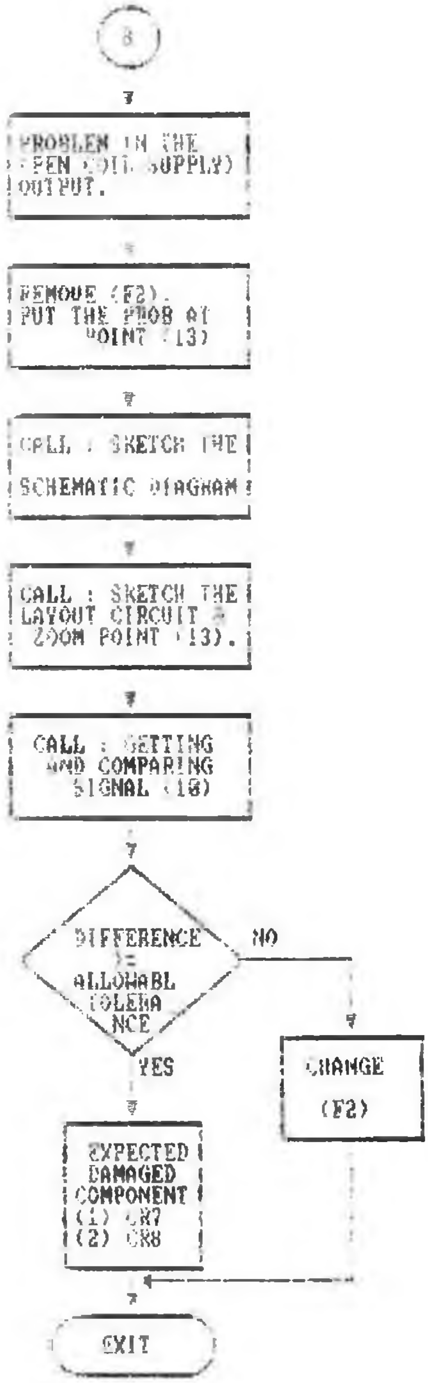
3
PROBLEM IN (-12V) OUTPUT, BETWEEN POINT(1)&POINT(5)
DISCONNECT (R23) AT POINT(6). PUT THE PROB AT THE FREE EDGE TO (R23).
CALL : SKETCH THE CIRCUIT & ZOOM POINT (6).
CALL : GETTING AND COMPARING SIGNAL (6)
DIFFERENCE >= ALLOWABL TOLERA NCE
YES
NO
4
EXPECTED DAMAGED COMPONENTS :
(1) C36,C38,C39,C40 C41,C42.
(2) CR13,CR14,CR15 CR16
EXIT

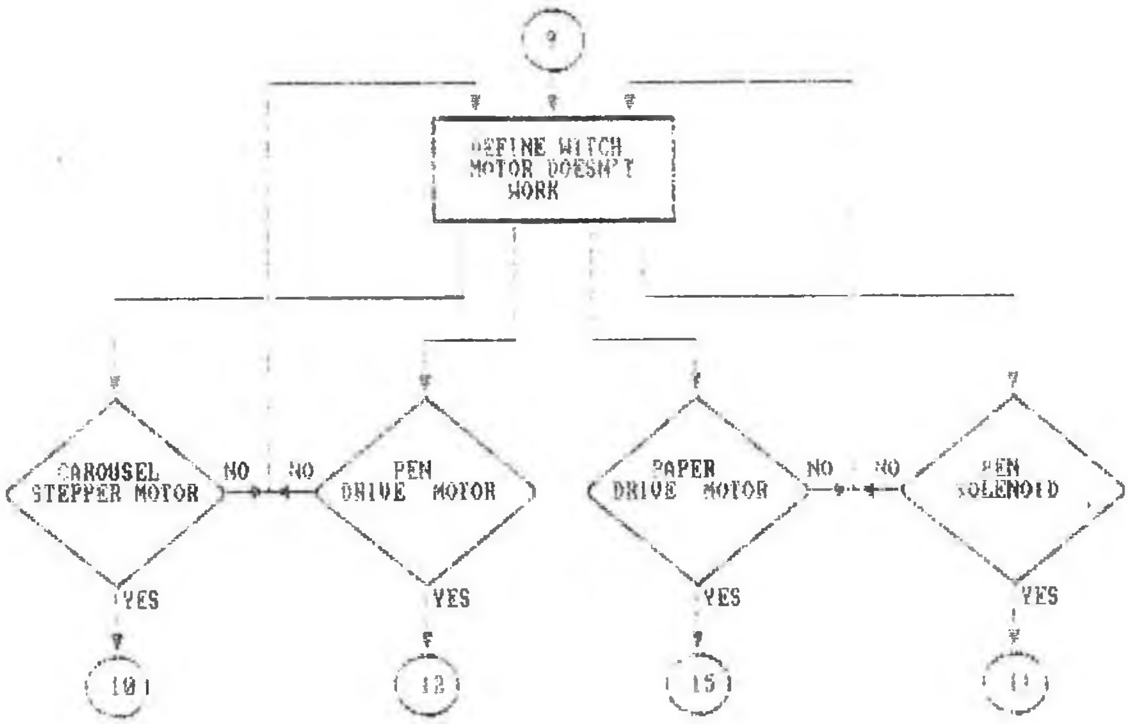

4
PROBLEM BETWEEN POINT(6).AND(-12V OR -5V) OUTPUT.
RECONNECT (R23). DISCONNECT(V21)AT POINT(6). PUT THE PROB AGAIN AT POINT(5).
CALL : SKETCH THE CIRCUIT & ZOOM POINT (5).
CALL : GETTING AND COMPARING SIGNAL (5).
YES
DIFFERENCE >= ALLOWABL TOLERA NCE
NO
EXPECTED DAMAGED COMPONENTS
(1) C31.C14
(2) VR3.VR4
EXIT
EXPECTED DAMAGED COMPONENTS
(1) C3
(2) VR2
EXIT


5
PUT THE PROB AT
POINT (8)
CALL : SKETCH
THE CIRCUIT &
ZOOM POINT (8).
CALL : GETTING
AND COMPARING
SIGNAL (8)
DIFFERENCE
<=
ALLOWABL
TOLERA
NCE
YES
6
NO
7

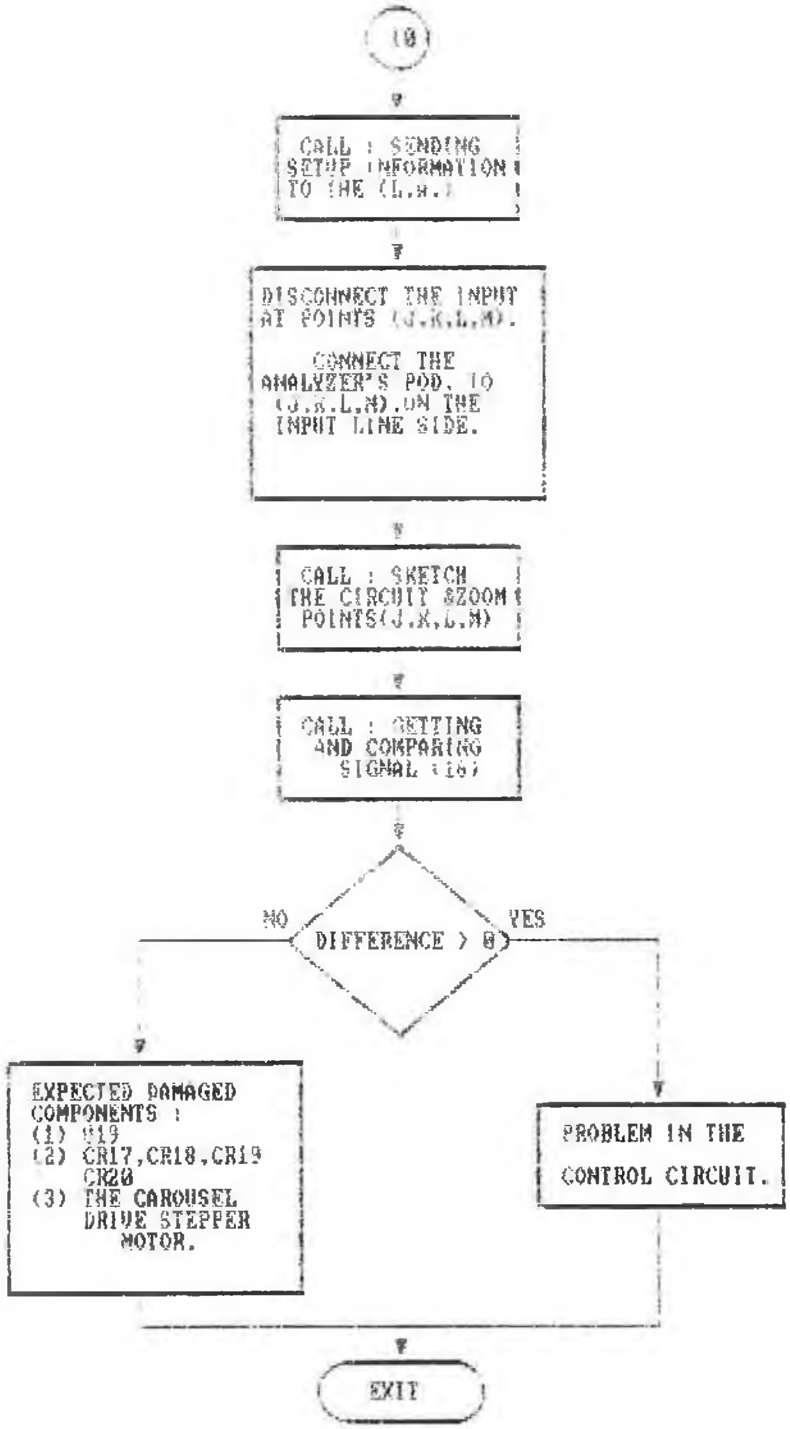

5
PROBLEM BETWEEN POINT(8)&POINT(1)
REMOVE (F1). PUT THE PROB AT POINT (9).
CALL : SKETCH THE CIRCUIT & ZOOM POINT (9)
CALL : GETTING AND COMPARING SIGNAL (8)
DIFFERENCE >= ALLOWABL TOLERANCE
NO
CHANGE (F1)
EXIT
YES
DISCONNECT (R20)AT POINT (10).PUT THE PROB AT THE SAME POINT.
CALL : SKETCH THE CIRCUIT & ZOOM POINT (10).
CALL : GETTING AND COMPARING SIGNAL (8)
DIFFERENCE >= ALLOWABL TOLERANCE
YES
EXPECTED DAMAGED COMPONENTS
(1) CR9,CR10
(2) CR11
(3) CR12,C2
EXIT
NO
EXPECTED DAMAGED COMPONENTS
(1) VR5,VR6
(2) C33
(3) C4,17
EXIT

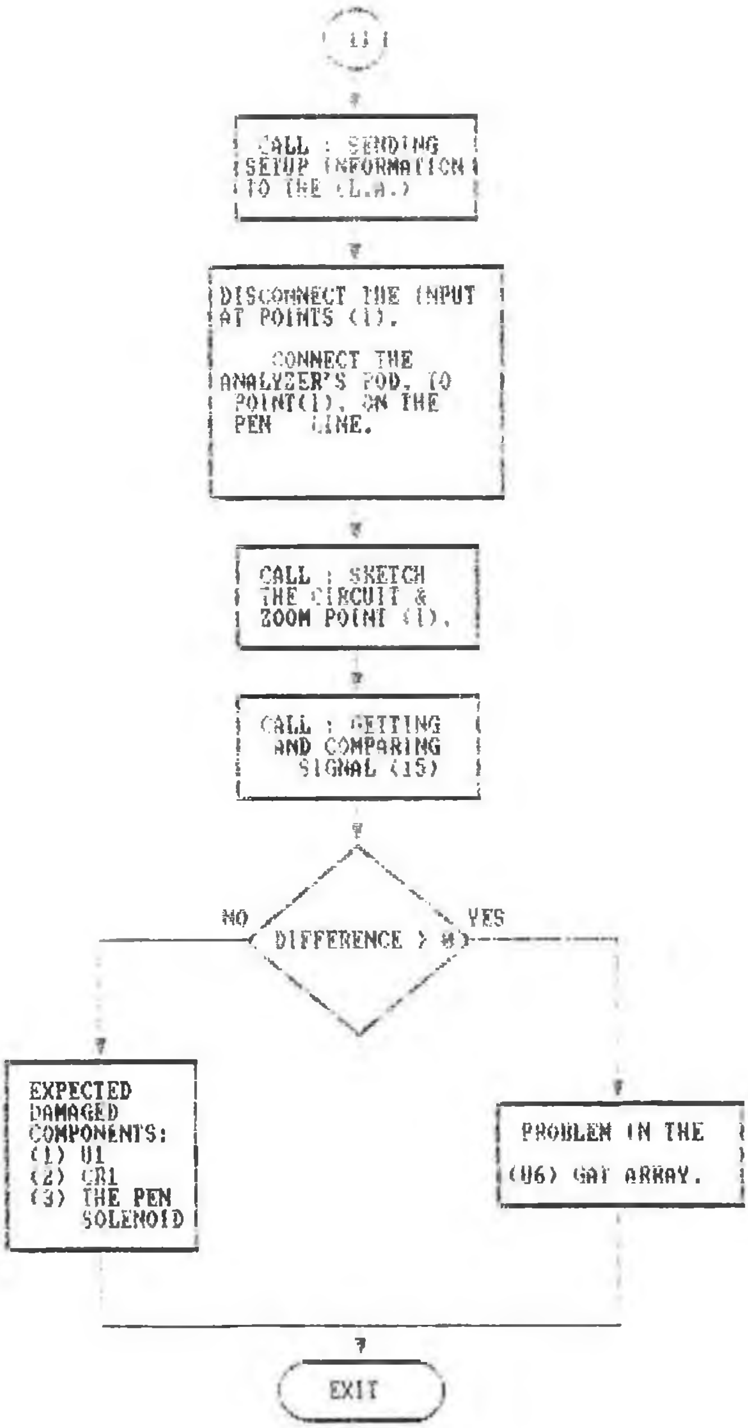

7
PUT THE PROB AT POINT (11)
CALL : SKETCH THE CIRCUIT & ZOOM POINT (11).
CALL : GETTING AND COMPARING SIGNAL (9)
DIFFERENCE >= ALLOWABL TOLERA NCE
YES
EXPECTED DAMAGED COMPONENTS (1) VR5,VR6 (2) C33 (3) C4,17
EXIT
NO
PUT THE PROB AT POINT (12)
CALL : SKETCH THE CIRCUIT & ZOOM POINT (12).
CALL : GETTING AND COMPARING SIGNAL (10)
DIFFERENCE >= ALLOWABL TOLERA NCE
YES
8
NO
RUN THE ENTERNAL TEST PROGRAM TO THE PLOTTER. WATCH THE PLOTTER MOTORS CAREFULY.
ALL MOTORS WORK
NO
9
YES
EXIT


8
PROBLEM IN THE (PEN COIL SUPPLY) OUTPUT.
REMOVE (F2). PUT THE PROB AT POINT (13)
CALL : SKETCH THE SCHEMATIC DIAGRAM
CALL : SKETCH THE LAYOUT CIRCUIT & ZOOM POINT (13).
CALL : GETTING AND COMPARING SIGNAL (10)
DIFFERENCE >= ALLOWABL TOLERA NCE
NO
YES
CHANGE (F2)
EXPECTED DAMAGED COMPONENT (1) CR7 (2) CR8
EXIT

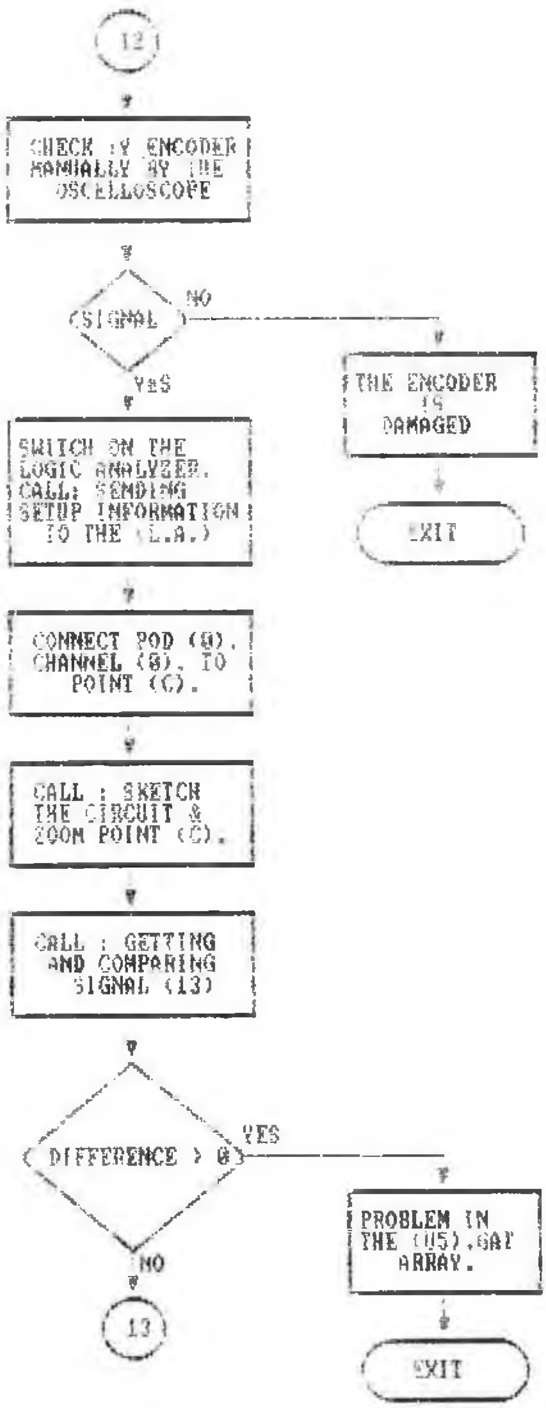


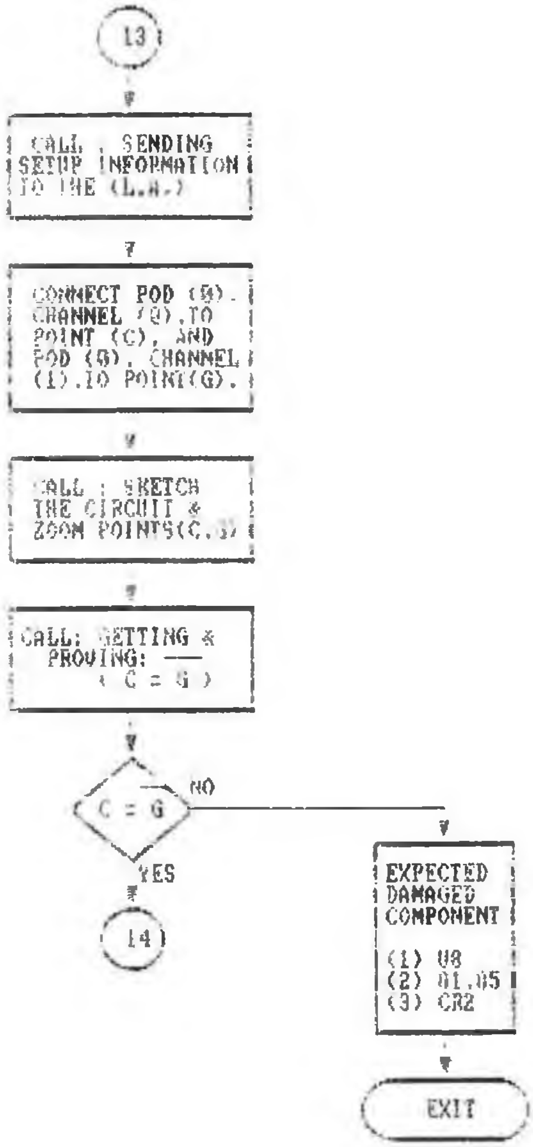
9
DEFINE WITCH MOTOR DOESN'T WORK
CAROUSEL STEPPER MOTOR
NO
NO
PEN DRIVE MOTOR
PAPER DRIVE MOTOR
NO
NO
PEN SOLENOID
YES
YES
YES
YES
10
12
15

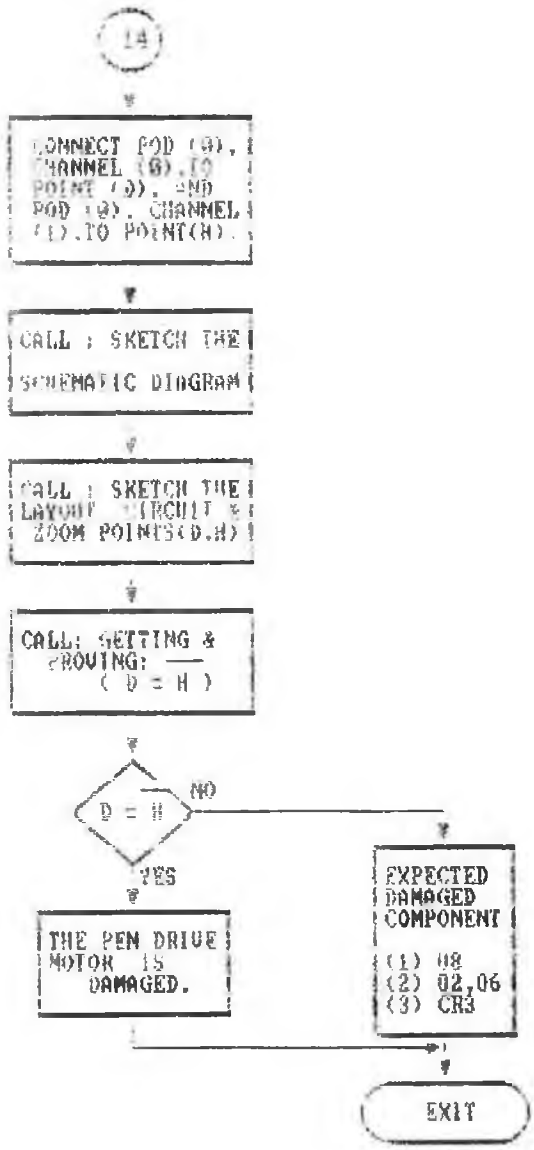

10
CALL : SENDING SETUP INFORMATION TO THE (L.A.)
DISCONNECT THE INPUT AT POINTS (J.K.L.M).
CONNECT THE ANALYZER'S POD, TO (J.K.L.M).ON THE INPUT LINE SIDE.
CALL : SKETCH THE CIRCUIT &ZOOM POINTS(J.K.L.M)
CALL : GETTING AND COMPARING SIGNAL (16)
DIFFERENCE > 0
NO
YES
EXPECTED DAMAGED COMPONENTS :
(1) U19
(2) CR17,CR18,CR19 CR20
(3) THE CAROUSEL DRIVE STEPPER MOTOR.
PROBLEM IN THE CONTROL CIRCUIT.
EXIT



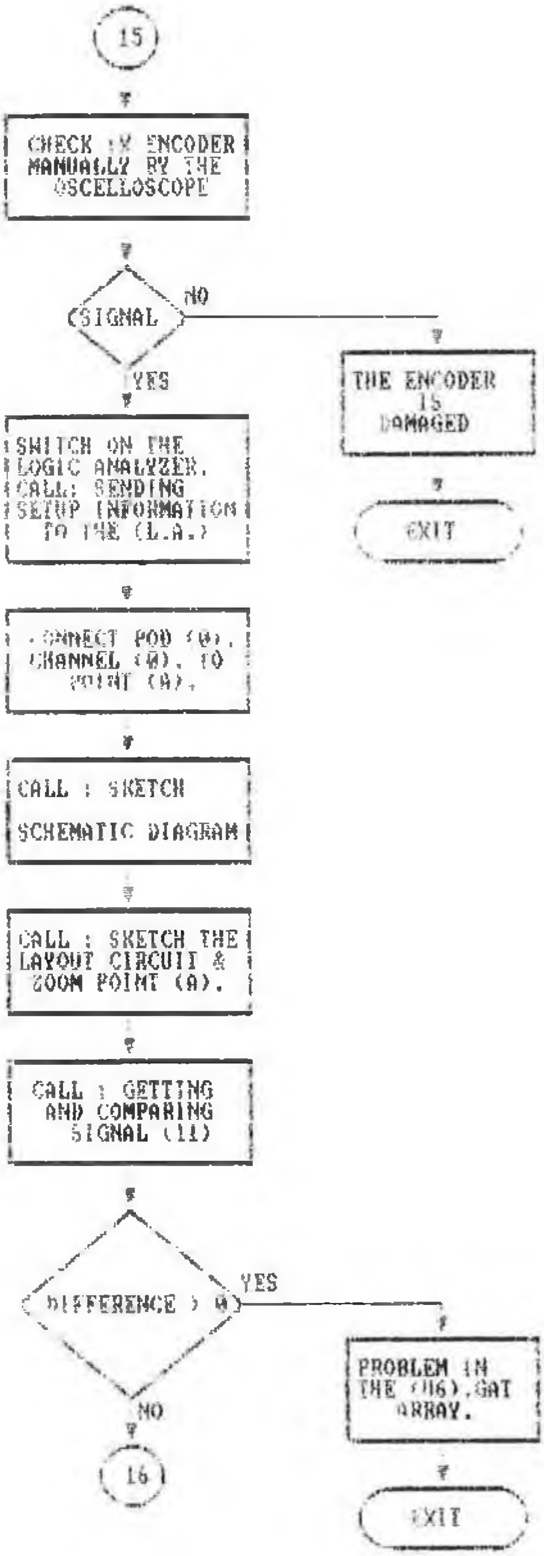
11
CALL : SENDING SETUP INFORMATION TO THE (L.A.)
DISCONNECT THE INPUT AT POINTS (1).
CONNECT THE ANALYZER'S POD, TO POINT(1), ON THE PEN LINE.
CALL : SKETCH THE CIRCUIT & ZOOM POINT (1).
CALL : GETTING AND COMPARING SIGNAL (15)
DIFFERENCE > 0
NO
YES
EXPECTED DAMAGED COMPONENTS:
(1) U1
(2) CR1
(3) THE PEN SOLENOID
PROBLEM IN THE (U6) GAT ARRAY.
EXIT

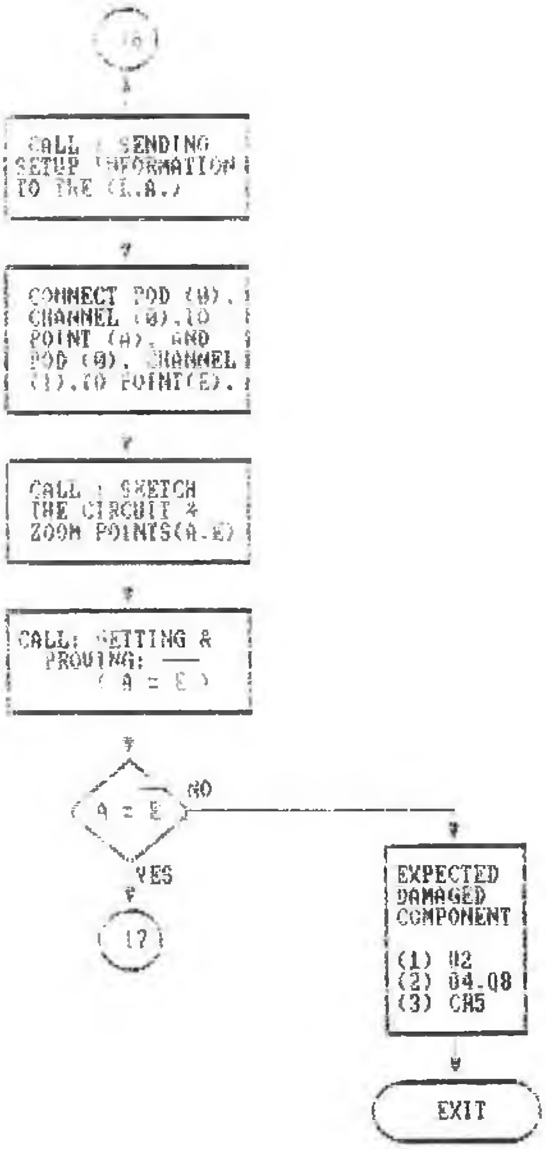

12
CHECK BY ENCODER MANUALLY BY THE OSCELLOSCOPE
SIGNAL
NO
YES
THE ENCODER IS DAMAGED
EXIT
SWITCH ON THE LOGIC ANALYZER. CALL: SENDING SETUP INFORMATION TO THE (L.A.)
CONNECT POD (0). CHANNEL (0). TO POINT (C).
CALL : SKETCH THE CIRCUIT & ZOOM POINT (C).
CALL : GETTING AND COMPARING SIGNAL (13)
DIFFERENCE > 0
YES
PROBLEM IN THE (U5).GAT ARRAY.
EXIT
NO
13

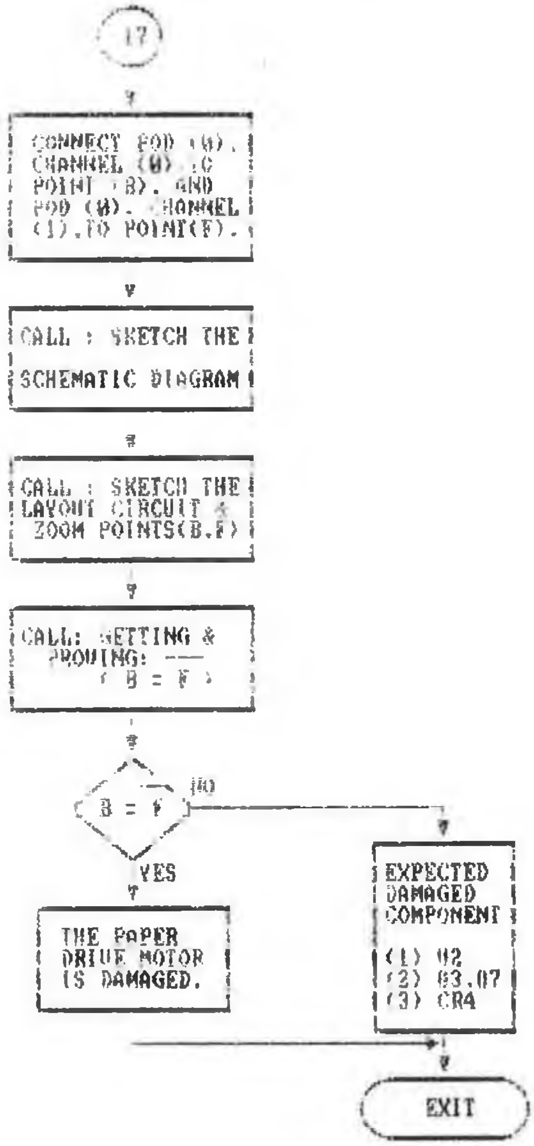

13
CALL : SENDING SETUP INFORMATION TO THE (L.A.)
CONNECT POD (0), CHANNEL (0).TO POINT (C), AND POD (0), CHANNEL (1).TO POINT(G).
CALL : SKETCH THE CIRCUIT & ZOOM POINTS(C,G)
CALL: GETTING & PROVING: (C = G)
C = G
NO
YES
14
EXPECTED DAMAGED COMPONENT
(1) U8
(2) Q1,Q5
(3) CR2
EXIT



14
CONNECT POD (0), CHANNEL (0) TO POINT (D), AND POD (0), CHANNEL (1) TO POINT(H).
CALL : SKETCH THE SCHEMATIC DIAGRAM
CALL : SKETCH THE LAYOUT CIRCUIT & ZOOM POINTS(D,H)
CALL: GETTING & PROVING: (D = H)
D = H
NO
YES
THE PEN DRIVE MOTOR IS DAMAGED.
EXPECTED DAMAGED COMPONENT
(1) Q8
(2) Q2,Q6
(3) CR3
EXIT


15
CHECK (X ENCODER MANUALLY BY THE OSCELLOSCOPE
SIGNAL
NO
YES
THE ENCODER IS DAMAGED
EXIT
SWITCH ON THE LOGIC ANALYZER. CALL: SENDING SETUP INFORMATION TO THE (L.A.)
CONNECT POD (0), CHANNEL (0), TO POINT (A).
CALL : SKETCH SCHEMATIC DIAGRAM
CALL : SKETCH THE LAYOUT CIRCUIT & ZOOM POINT (A).
CALL : GETTING AND COMPARING SIGNAL (11)
DIFFERENCE > 0
YES
NO
PROBLEM IN THE (U6).GAT ARRAY.
EXIT
16


16
CALL : SENDING SETUP INFORMATION TO THE (L.A.)
CONNECT POD (0), CHANNEL (0),TO POINT (A), AND POD (0), CHANNEL (1),TO POINT(E).
CALL : SKETCH THE CIRCUIT & ZOOM POINTS(A-E)
CALL: SETTING & PROVING: (A = E)
(A = E)
NO
YES
17
EXPECTED DAMAGED COMPONENT
(1) U2
(2) Q4-Q8
(3) CR5
EXIT



17
CONNECT POD (0). CHANNEL (0) TO POINT (B). AND POD (0). CHANNEL (1).TO POINT(F).
CALL : SKETCH THE SCHEMATIC DIAGRAM
CALL : SKETCH THE LAYOUT CIRCUIT & ZOOM POINTS(B.F)
CALL: SETTING & PROVING: (B = F)
B = F
NO
YES
THE PAPER DRIVE MOTOR IS DAMAGED.
EXPECTED DAMAGED COMPONENT
(1) Q2
(2) Q3.Q7
(3) CR4
EXIT