

MICROPROCESSOR BASED IC TESTERS

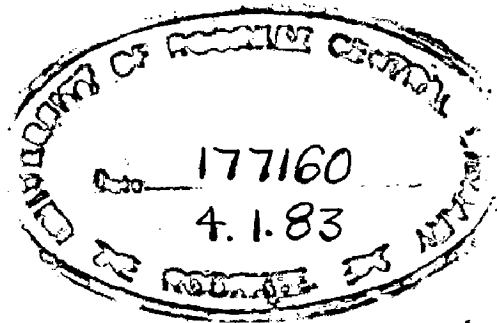
A DISSERTATION

Submitted in partial fulfilment of the
requirements for the award of the degree
of
MASTER OF ENGINEERING
(System Engineering & Operations Research)

by

Capt H. C. LOHUMI

CHECKED
1983



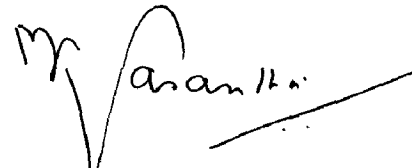
DEPARTMENT OF ELECTRICAL ENGINEERING
UNIVERSITY OF ROORKEE
ROORKEE-247 672 (U. P.)

Oct. 1982

C E R T I F I C A T E

Certified that the dissertation entitled "MICROPROCESSOR BASED IC TESTERS" which is being submitted by Capt H C LOHUMI in partial fulfilment for the award of the Degree of MASTER OF ENGINEERING in ELECTRICAL ENGINEERING (System Engineering & Operations Research) of the University of Roorkee, Roorkee, is a record of student's own work carried out by him under my supervision and guidance. The matter embodied in this dissertation has not been submitted for the award of any other Degree or Diploma.

This is further to certify that he has worked for a period of about nine months from Jan.82 ~ to Oct.82 ~ for preparing this dissertation at this University.



(M.K. VASANTHA)
Reader

Electrical Engg. Department
University of Roorkee
ROORKEE-247 672 U.P.

Dated : Oct.13, 1982

A C K N O W L E D G E M E N T

I would like to express my gratitude to Mr.M.K.Vasantha, Reader, Department of Electrical Engineering, University of Roorkee, Roorkee for initiating me in the field of microprocessor applications implementation. I am, indeed, indebted to him for his unflagging support and invaluable guidance in solving numerous problems encountered in the practical work.

I am also grateful to Dr. G.K.Tandon, Head, Computer Centre for permitting me the use of ADM-3A terminal for this work.

I am extremely grateful to Mr.M.Pant, Lecturer, Department of Electrical Engineering and Mr. A.K.Raja, Reader, Department of Electrical Engineering for permitting ████ unrestricted access to various facilities at their disposal, freely and for material assistance.

Special thanks are due to Mr. L.K.Gupta for helping me in solving the problems encountered in PCB design and fabrication. Lastly, I would like to thank all the teachers and the staff of the Department for assisting me in various stages of this dissertation.


Capt H C Lohumi

A B S T R A C T

Microprocessor based systems are being used increasingly in nearly all the walks of life. Their usage is wide spread, from house-hold gadgets to sophisticated missiles. In order to be able to design even simple microprocessor based systems, it is necessary to understand completely the working of the microprocessor chip and allied peripherals. For this, a 'hand on' experience is a must. The aim of this dissertation was to get this experience. Microprocessor based digital IC testers were a natural choice of the practical system selected for design because of the necessity of testing IC chips before actually using them.

The basic unit used was HIL-2961 microprocessor trainer, an INTEL 8085A based system. A brief description of this unit is given in Chapter 1. Chapter 2 deals with the underlying principles of IC testing alongwith an initial test programme for IC 7400.

A CRT terminal is a very useful peripheral device for increasing the user efficiency in communicating with the microcomputer. The IC tester developed in this dissertation operates through ADM-3A, a CRT terminal.

Abstract contd..

Chapter 3 deals with design of the hardware interface between the terminal and the microcomputer. Monitor developed for the IC tester is discussed in Chapter 4.

Test programmes for various ICs are discussed in Chapter 5 and finally the design of a Universal IC Tester is discussed in Chapter 6.

..

| | | | |
|---|--|----|------|
| 1 | GETTING ACQUAINTED WITH HIL-2961 MICROPROCESSOR TRAINER | 11 | 1-1 |
| | 1.1 Introduction | 11 | 1-1 |
| | 1.2 Clock Frequency | 11 | 1-3 |
| | 1.3 Memory Allocation and Decoding Circuitry | 11 | 1-3 |
| | 1.4 Input and Output Ports | 11 | 1-5 |
| | 1.5 Keyboard Management | 11 | 1-9 |
| | 1.6 Bus Protection | 11 | 1-9 |
| | 1.7 Utility Programmes | 11 | 1-11 |
| 2 | GENESIS OF IC TESTING AND 7400 TEST PROGRAMME | 11 | 2-1 |
| | 2.1 Introduction | 11 | 2-1 |
| | 2.2 Detection of Logic Malfunction | 11 | 2-3 |
| | 2.3 Implementation of Test Sets | 11 | 2-5 |
| | 2.4 Development of IC Tester | 11 | 2-7 |
| | 2.5 Quadruple Two Input Nand Gate IC 7400 | 11 | 2-8 |
| | 2.6 Minimal Test Set Development for IC 7400 | 11 | 2-8 |
| | 2.7 Key Debouncing | 11 | 2-10 |
| | 2.8 Programme for IC 7400 Test | 11 | 2-13 |
| 3 | INTERFACING HIL-2961 WITH CRT TERMINAL | 11 | 3-1 |
| | 3.1 Introduction : Need for a CRT Terminal | 11 | 3-1 |
| | 3.2 Interactive Display Terminal ADM-3A | 11 | 3-2 |
| | 3.3 Development of Interface | 11 | 3-4 |
| | 3.4 Hardware Interface | 11 | 3-5 |
| | 3.5 Power Supply | 11 | 3-6 |
| | 3.6 Baud Rate Generator | 11 | 3-6 |

| | | | |
|---|--|----|------|
| | 3.6.1 Design Calculations | .. | 3-8 |
| | 3.6.2 Operating the Baud Rate Generator | .. | 3-9 |
| | 3.7 RS-232C and 8251A USART Interface | .. | 3-11 |
| | 3.7.1 INTEL 8251A USART | .. | 3-14 |
| 4 | MONITOR FOR THE CRT TERMINAL | .. | 4-1 |
| | 4.1 Introduction | .. | 4-1 |
| | 4.2 Sign on Message Programme | .. | 4-2 |
| | 4.3 Get Command Programme | .. | 4-2 |
| | 4.4 Functional Commands | .. | 4-3 |
| | 4.5 Try Command | .. | 4-4 |
| | 4.6 List Command | .. | 4-5 |
| | 4.7 Execute (GO) Command | .. | 4-5 |
| | 4.8 Subroutines for Programmes | .. | 4-6 |
| | 4.9 Monitor Listing | .. | 4-6 |
| 5 | TEST PROGRAMMES DEVELOPMENT FOR A FEW SELECTED IC CHIPS | .. | 5-1 |
| | 5.1 Introduction | .. | 5-1 |
| | 5.2 Test Programmes | .. | 5-1 |
| | 5.3 Testing Procedure | .. | 5-2 |
| | 5.4 IC 7400 Test Programme | .. | 5-4 |
| | 5.5 IC 7476 Test Programme | .. | 5-4 |
| | 5.6 IC 7490 Test Programme | .. | 5-6 |
| | 5.7 IC 7493 Test Programme | .. | 5-9 |

| | | | |
|---|---|----|-----|
| 6 | UNIVERSAL IC TESTER | .. | 6-1 |
| | 6.1 Introduction | .. | 6-1 |
| | 6.2 Requirements | .. | 6-1 |
| | 6.3 Design of Universal IC Tester | .. | 6-2 |
| | 6.4 Programming the Universal IC Tester | .. | 6-4 |
| | 6.5 Comments | .. | 6-4 |
| 7 | CONCLUSION | .. | 7-1 |
| | 7.1 Summary of the Work | .. | 7-2 |
| | 7.2 Recommended Developments | .. | 7-2 |
| | REFERENCES | .. | R-1 |

APPENDICES

Appendix No.

| | | | |
|---|---|----|-----|
| A | INTEL 8085A : Brief Description | .. | A-1 |
| B | 8085A Instruction Set | .. | B-1 |
| C | HIL-2961 Utility Programmes : Explanation | .. | C-1 |
| D | Interactive Display Terminal ADM-3A | .. | D-1 |
| E | Details of Components | .. | E-1 |
| F | Serial Data Transmission Formats | .. | F-1 |
| G | INTEL 8251A Universal Synchronous Asynchronous Transmitter Receiver | .. | G-1 |
| H | Initialization Programme for Universal IC Tester Testing IC 7400 | .. | H-1 |

CHAPTER - 1

GETTING ACQUAINTED WITH HIL-2961 MICROPROCESSOR TRAINER

1.1 INTRODUCTION

HIL-2961 microprocessor trainer (1), marketed by Hindustan Instruments Limited, is a microcomputer based on INTEL's 8085A μ P. This system formed the basis of all the work carried out in this dissertation. It is, therefore, necessary to gain an insight into the various functional aspects of this system before proceeding further with the discussion of the dissertation work.

The pin configuration and brief description of the 8085A μ P is given at Appendix 'A'. The summary of the instruction set is given at appendix 'B'. Fig.1.1 shows a block diagram representation of HIL-2961's architecture. The user can communicate with the system through a 24 key keyboard. Sixteen of these keys are used for the Hexa decimal code and the remaining eight keys viz., FETCH REG, FETCH ADDR, STORE/INC, DECR, SINGLE STEP, FETCH PC, MC STEP and EXECUTE provide the user the facility of entering and executing his programme as well as facilitate software debugging.

Results of various operations are displayed on a six digit seven segment display provided on the top right hand side of the trainer. The first four digits show the addressed location

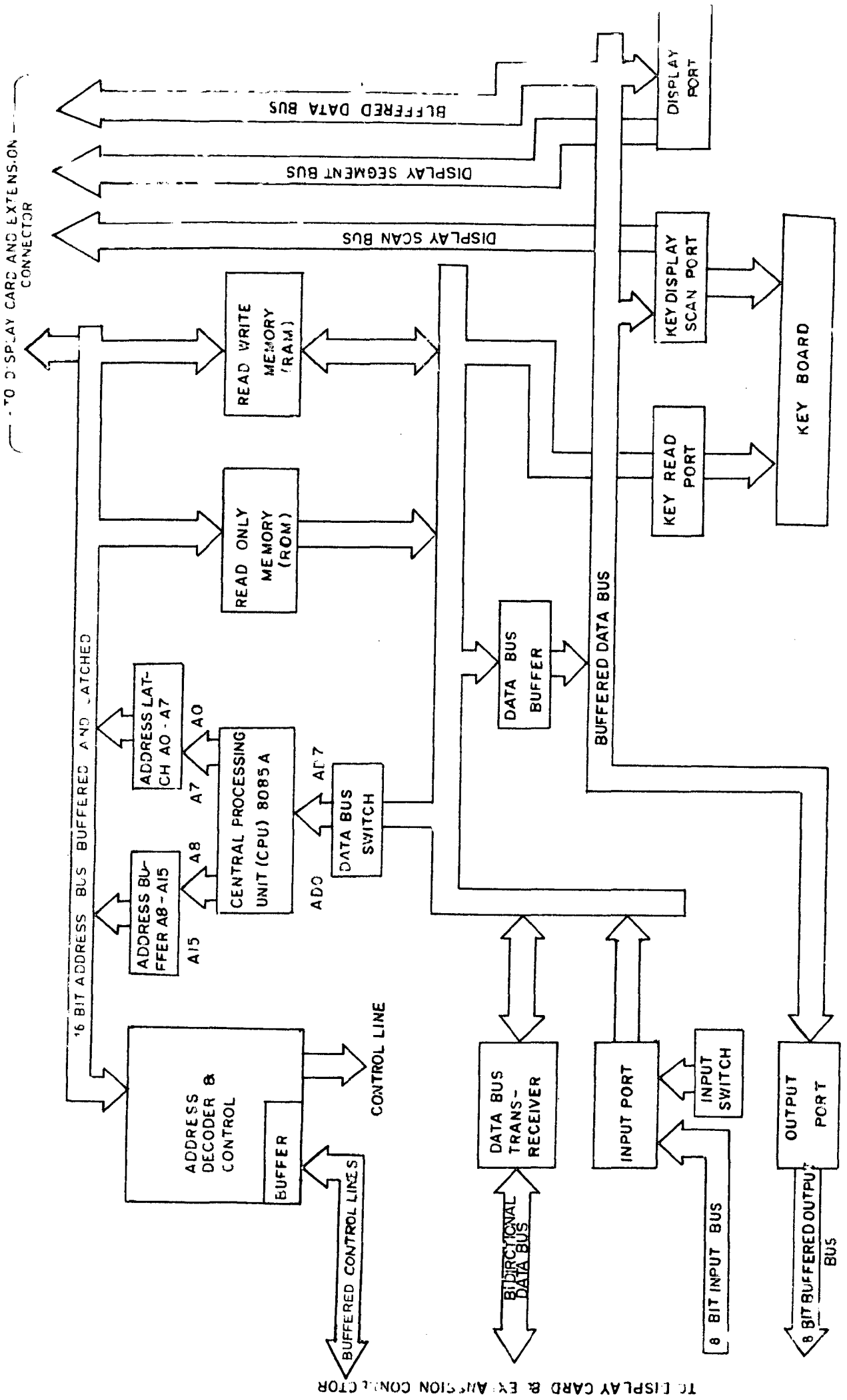


FIG. 1 HL-2961 ARCHITECTURE.

in hexa-decimal code and the last two display the content of that location. This display is software managed. Another six digit, seven segment display provided on the top left hand side of the trainer displays the address and its content in a similar manner in machine step operation. This display is hardware managed.

1.2 CLOCK FREQUENCY

The 8085 A CPU can operate at a maximum internal clock frequency of 3.125 MHz. The actual operating frequency is dependent upon the parallel resonant frequency of the quartz crystal placed at its X_1 and X_2 inputs, which must be twice the internal frequency desired. Thus the 8085 A CPU can accommodate a quartz crystal having a parallel resonant frequency of 6.25 MHz or less. In HIL-2961, the external quartz crystal has the parallel resonant frequency of 4.00 MHz, thereby generating an internal clock frequency of 2.00 MHz for the CPU operation. Practically, this frequency was measured using a digital frequency counter and found to be 2.0007 MHz.

1.3 MEMORY ALLOCATION AND DECODING CIRCUITRY

The 8085A μ P can address 64K bytes of memory. Of these, only 16 K bytes have been decoded in the HIL 2961. This gives approximately 1.8 K bytes of memory space to the user. Fig.1.2 shows the system address map for various devices used in the trainer. The memory allocation for the RAM, ROM, one I/P port

| BIT | UPPER HALF OF ADDRESS IN BINARY | | | | | | ADDRESS IN HEX | DEVICE |
|-----|------------------------------------|----|----|----|----|--------|-------------------|-----------------|
| | 15 | 14 | 13 | 12 | 11 | 10 9 8 | | |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 0 0 0 | ROM |
| | 0 | 0 | 0 | 0 | 0 | 1 1 1 | 0 7 F F | |
| | 0 | 0 | 0 | 0 | 1 | 0 | 0 0 0 | RAM |
| | 0 | 0 | 0 | 0 | 1 | 1 1 1 | 0 F F F | |
| | 0 | 0 | 0 | 1 | 0 | 0 | 0 0 0 | KEY DATA |
| | 0 | 0 | 0 | 1 | 1 | 0 | 1 7 F F | |
| | 0 | 0 | 0 | 1 | 1 | 0 | 0 0 0 | INPUT PORT |
| | 0 | 0 | 1 | 0 | 0 | 0 | 2 0 0 0 | |
| | 0 | 0 | 1 | 0 | 0 | 1 | 1 1 1 | SCAN |
| | 0 | 0 | 1 | 0 | 1 | 0 | 0 0 0 | |
| | 0 | 0 | 1 | 0 | 1 | 1 | 1 1 1 | OUTPUT PORT |
| | 0 | 0 | 1 | 0 | 1 | 0 | 2 7 F F | |
| | 0 | 0 | 1 | 1 | 0 | 0 | 0 0 0 | DISPLAY SEGMENT |
| | 0 | 0 | 1 | 1 | 1 | 0 | 0 0 0 | |
| | 0 | 0 | 1 | 1 | 1 | 1 | 1 1 1 | NOT USED |
| | 0 | 1 | 0 | 0 | 0 | 0 | 0 0 0 | |
| | 1 | 1 | 1 | 1 | 1 | 1 | F F F F | |

16 K ADDRESS LOCATIONS

48 K ADDRESS LOCATIONS

MEMORY

I/O

FIG.1.2 SYSTEM ADDRESS MAP FOR 2961 MICROPROCESSOR TRAINER

| BIT | UPPER HALF OF ADDRESS IN BINARY | | | | | | ADDRESS IN HEX | | DEVICE | |
|-----------------------------------|------------------------------------|----|----|----|----|----|-------------------|---------|---------|--------------------|
| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | | |
| ↑ 16 K ADDRESS LOCATIONS | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 0 0 0 | ROM |
| | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 7 F F | |
| | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 8 0 0 | RAM |
| | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 F F F | |
| | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 0 0 0 | KEY DATA |
| | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 7 F F | |
| | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 8 0 0 | INPUT PORT |
| | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 F F F | |
| | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 2 0 0 0 | SCAN |
| | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 2 7 F F | |
| | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 2 8 0 0 | OUTPUT PORT |
| | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 2 F F F | |
| | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 3 0 0 0 | DISPLAY SEGMENT |
| | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 3 7 F F | |
| | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 3 8 0 0 | NOT USED |
| | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 3 F F F | |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 4 0 0 0 | | |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | F F F F | | |

FIG.1.2 SYSTEM ADDRESS MAP FOR 2961 MICROPROCESSOR TRAINER

and one O/P port' is shown in Fig.1.3. Memory locations from 2000H to 27FFH serve a dual purpose. This space may either be utilized for addressing the Input port provided on the trainer, or else it may be used for 2K ROM storing the software for audio cassette interface, EPROM programming and some utility programmes. Either of these functions may be selected by the user by suitable positioning of a jumper provided on the trainer PCB. The address decoding circuitry for the 16K memory space is shown in Fig.1.4.

1.4 INPUT AND OUTPUT PORTS

The microprocessor trainer provides an eight bit parallel input port and an eight bit parallel output port. These ports may be addressed either through in-structions based on I/O mapped I/O (i.e. IN Port or OUT port instructions) or those based on memory mapped I/O (i.e. STA addr or LDA addr and other memory reference instructions). Since each of these ports is allocated 2K of memory space in the decoding, these ports may be addressed by any of the addresses shown in Table 1.1.

Both the ports are available on the 60-pin extension connector provided on the trainer. Through this connector all the signals of the 8085A CPU as well as some control signals used in the 2961 architecture are made available to the user for any hardware development.

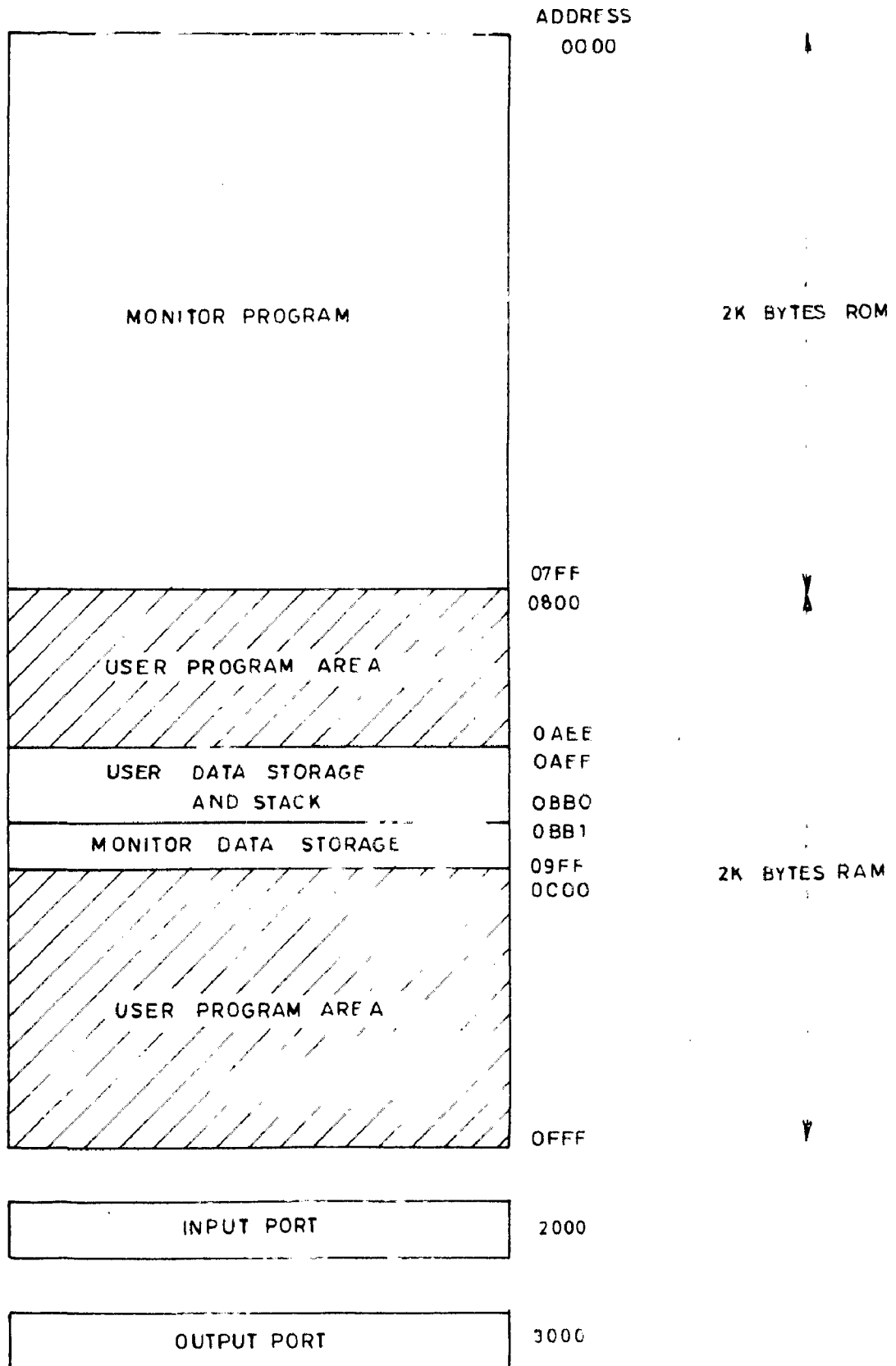


FIG.1.3 MEMORY ALLOCATION IN HIL 2961

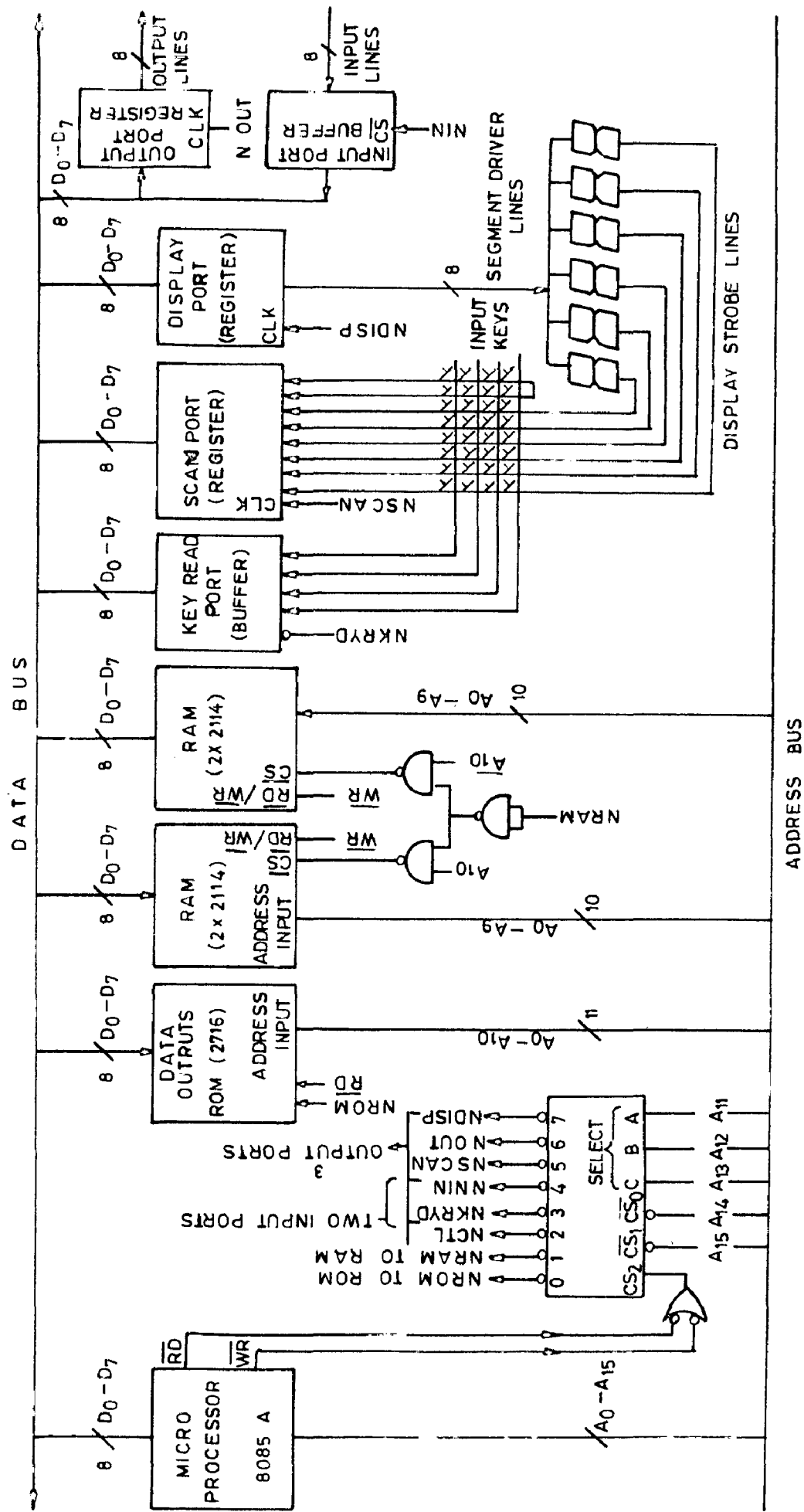


FIG.1.4 BLOCK DIAGRAM OF THE 2961 MICROPROCESSOR TRAINER SHOWING ADDRESS DECODING

TABLE 1.1 : ADDRESSES FOR PORT SELECTION

| S.No. | Port | Type of addressing | Memory addresses/Port Nos for selection. |
|-------|----------|--------------------|--|
| 1 | I/P port | I/O mapped | Port 20H to 27H |
| 2 | -do- | Memory mapped | 2000H to 27FFH |
| 3 | O/P port | I/O mapped | Port 30 to 37H |
| 4 | -do- | Memory mapped | 3000H to 37FFH |

The data on the output port can be read on a set of 8 LEDs labelled OUTPUT on the display pannel. This data is latched through 74LS273. This was very suitable for developing the IC tester, since a particular input for the IC under test could be held valid while the output of the IC was being read into the μ P through the Input port.

In contrast, the Input port data bus is tristate buffered. Data can be entered into the input port by means of 'INPUT' switches provided on the trainer PCB. In order to load the data on the input port externally, all the switches must be held in 'HI' position, which pulls all the eight bits to logical 1. In addition, the jumper mentioned in Sec. 1.3 must be appropriately placed for enabling communication with the CPU through the input port.

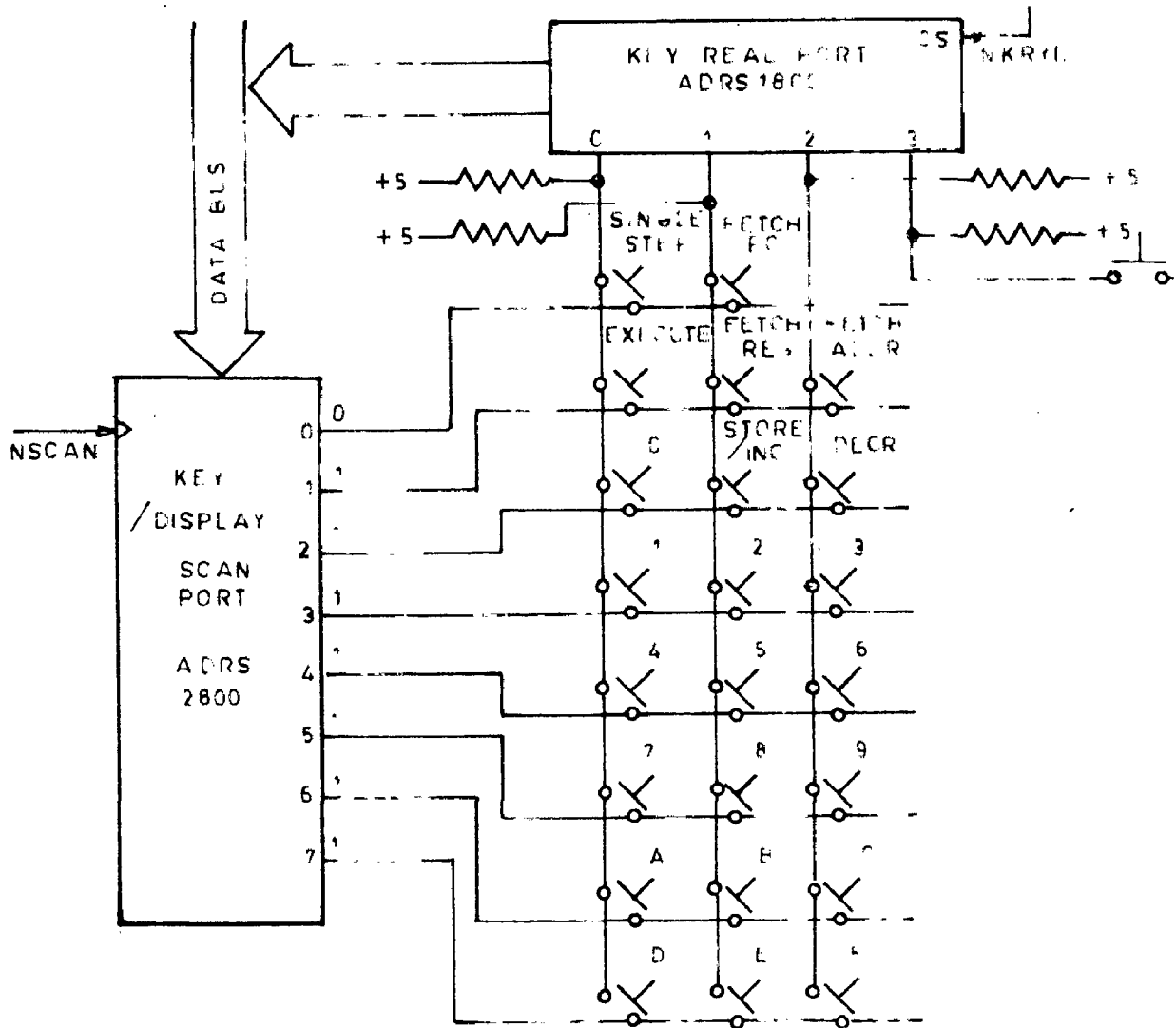


FIG 1.5 KEYBOARD INTERFACE IN HIL 2961

TABLE 1.1 : ADDRESSES FOR PORT SELECTION

| S.No. | Port | Type of addressing | Memory addresses/Port Nos for selection. |
|-------|----------|--------------------|--|
| 1 | I/P port | I/O mapped | Port 20H to 27H |
| 2 | -do- | Memory mapped | 2000H to 27FFH |
| 3 | O/P port | I/O mapped | Port 30 to 37H |
| 4 | -do- | Memory mapped | 3000H to 37FFH |

The data on the output port can be read on a set of 8 LEDs labelled OUTPUT on the display panel. This data is latched through 74LS273. This was very suitable for developing the IC tester, since a particular input for the IC under test could be held valid while the output of the IC was being read into the μ P through the Input port.

In contrast, the Input port data bus is tristate buffered. Data can be entered into the input port by means of 'INPUT' switches provided on the trainer PCB. In order to load the data on the input port externally, all the switches must be held in 'HI' position, which pulls all the eight bits to logical 1. In addition, the jumper mentioned in Sec. 1.3 must be appropriately placed for enabling communication with the CPU through the input port.

1.5 KEYBOARD MANAGEMENT

The keyboard consists of 24 keys. In order to differentiate between various keys, the keyboard is arranged into an 8 x 3 matrix. This is shown in Fig.1.5. The key scan port behaves as an output port and the key read port behaves as an input port. Through a monitor subroutine called KIND, different keys are identified uniquely as follows:

1. Data 11111110 is sent to the Scan port. Thus a LOW is placed on row zero, other rows remain HIGH.
2. The key read port reads the column information. Since all bits are pulled HIGH, the column information will be XXXX1111 if no key is pressed.
3. In case a particular key in row zero is pressed, that particular column will be set LOW. This will be detected while reading the key read port.
4. If no key is found pressed, the monitor will arrange to place a zero on row 2,3 & so on and follow the same procedure. When a particular key is found pressed, the μP branches to appropriate location connected with that key.

1.6 BUS PROTECTION

In order to protect the system from any conflict arising due to placement of data continuously by the user erroneously on the data bus, while the CPU is directed to perform other operations requiring the data bus, the trainer data bus,

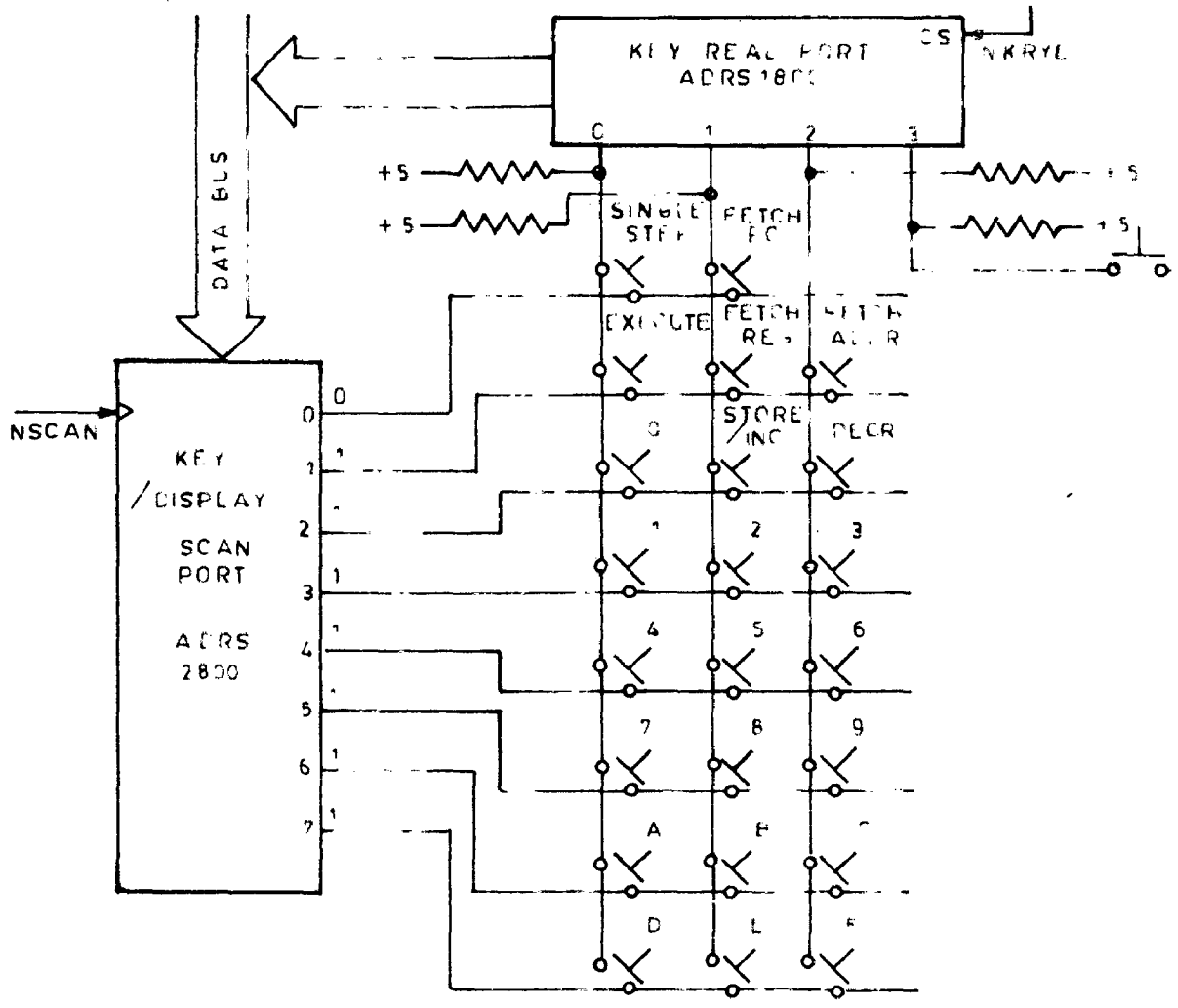


FIG. 1.5 KEYBOARD INTERFACE IN HIL 2961

available externally at the expansion connector, has been designed to remain in output mode, normally. The output buffers are permanently enabled by connecting the D OUT terminal to Ground. The input buffers are permanently disabled by connecting D IN terminals to +5V line directly. In case the user wants to expand the system, the data bus must be programmed to be in the desired input or output mode. This is achieved by first disconnecting the permanent connections of D OUT to Ground and of D IN to +5V and then connecting a logic circuit shown in Fig.1.6.

As mentioned in Section 1.3, only 16K bytes of memory space has been decoded inside the trainer. It can be seen in Fig.1.4 that A15 and A14 lines are always logical ZERO while addressing any location inside the trainer. For this address space, the circuit of Fig.1.6 ensures that the data bus is always in output mode irrespective of \overline{RD} signal. (D OUT is LOW and D IN is HIGH). When any location from 4000H to FFFFH is being addressed, either A15 or A14 or both will always be logical ONE and now the data bus will be in input mode whenever either \overline{RD} goes LOW or \overline{INTA} goes LOW. Otherwise, the data bus will remain in output mode.

1.7 UTILITY PROGRAMS

Two of the utility programmes viz., DELB and HORN subroutines have been incorporated in the test programmes developed for IC testing. Assembly language programmes for these subroutines

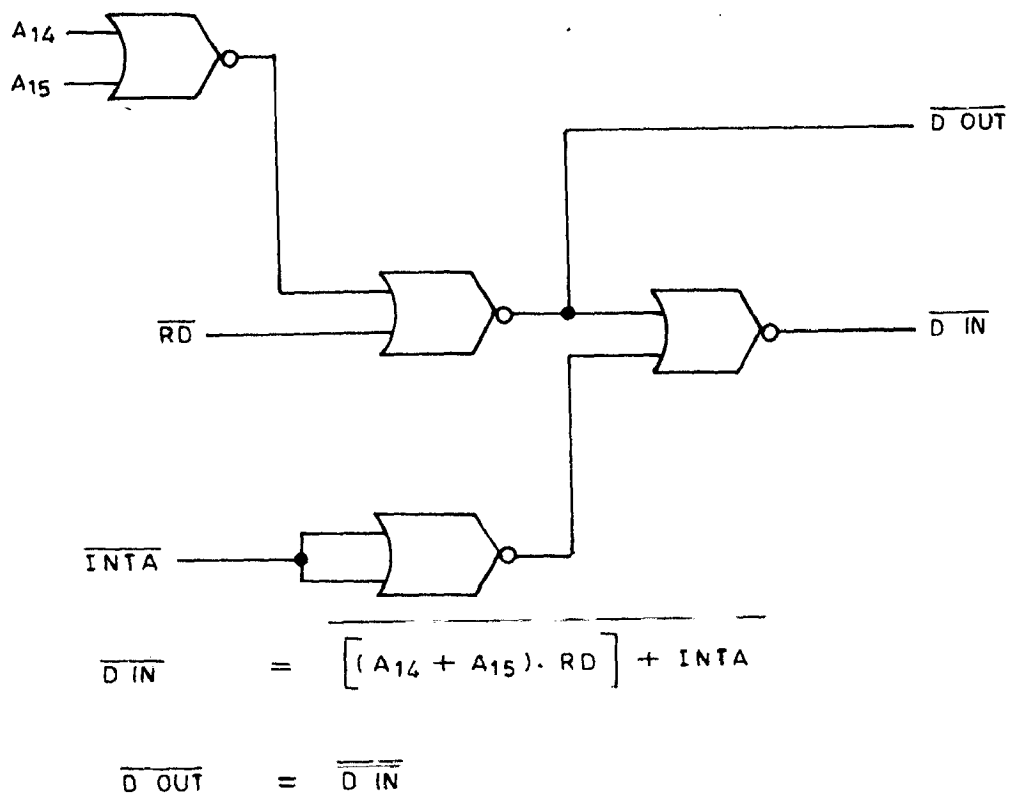


FIG. 1.6 PROGRAMMING THE HIL 2961 EXTERNAL DATA BUS

were written down from the flow charts provided in the users' manual of HIL-2961. These programmes are given at Appendix 'C'.

C H A P T E R - 2

GENESIS OF IC TESTING AND 7400 TEST PROGRAMME

2.1 INTRODUCTION

Integrated circuit technology and its applications have brought about revolutionary changes in the design of circuits and systems. IC chips enable tremendous reduction in overall system size, increase system reliability and reduce maintenance and repair problems. Put in a well designed circuit, a good IC gives a long and trouble free service. Due to the dependence of correct system behaviour on the correct behaviour of the IC chip, there is an obvious need for testing an IC chip before using it in a circuit. An IC chip has numerous behavioural aspects. These can be grouped, very generally, into two categories, viz., parametric behaviour and functional behaviour.

Parametric behaviour includes DC characteristics like threshold voltage, maximum voltage, fan-in, fan-out; dynamic characteristics like propagation delays,

transient response; and frequency sensitivity, specifying the frequency range over which the IC can perform the desired function. Functional behaviour, on the other hand, describes the output response (logic function) of the IC chip to the specified combinations of input signals.

In order to absolutely minimize the risk of circuit malfunction, all the behavioural characteristics of IC chips, used in the circuit, must be tested. An IC tester incorporating all the testing provisions, ^{must} by necessity, be extremely fast and economically viable, if it is to be of any practical use. Besides, the design of such a tester is very complex and is beyond the perview of this dissertation.

A designer of a digital circuit must have detailed knowledge of digital electronics and of the capacities and capabilities of different ICs used in the circuit. A good designer takes care to ensure that, in so far as possible, the IC does not operate in a critical zone and that sufficient safety margin exists. Also, the production process of IC chips incorporates sufficient safeguards to guarantee very high overall reliability (95-98%).

Most irksome and at times, the most frequently encountered hazard is the spurious IC chip. Spurious chips are, therefore, responsible for major share of overall chip malfunctions. Such an IC chip can be easily spotted by functional testing. It can thus be seen that, provided sufficient care has been taken to ensure sufficient margin in parametric capabilities, an IC chip tested for correct functional behaviour, will function correctly in the circuit, in overwhelming majority of cases.

2.2 DETECTION OF LOGIC MALFUNCTION

Digital circuits are basically classified into combinational and sequential circuits. In combinational circuits, the circuit outputs depend entirely upon the present set of input vectors. In sequential circuits, the circuit outputs depend not only upon the present set of input vectors but also upon the past history of inputs. In either category,

a logic malfunction may be permanent or time varying. Permanent faults relate to faults caused by a particular line permanently stuck at zero (s-a-0) or stuck at one (s-a-1). Time varying or intermittent faults are caused by very close tolerances in the chip or due to general deterioration of components.

This basic difference leads to different approaches for developing fault detection test sets for combinational and sequential circuits. Without going into further details, it will suffice to say that in combinational circuits a minimal test set may be found by using only those input combinations for which the output function is different for correct and faulty circuit behaviour(2 , 3 , 4).

In development of test sets for sequential circuits, the emphasis is more on finding a suitable procedure for ensuring fault detection. Usually, the first step is to identify the current state of the circuit or to bring it to a known state and then apply test inputs in the sequence for which the output sequence is known.

Needless to say, the IC chips implementing combinational and sequential circuits will need accordingly specified testing procedures.

2.3 IMPLEMENTATION OF TEST SETS

Having developed the test set for an IC, the problem is confined to developing means of applying the test inputs and observing the responses. Manual procedures are discarded since the time factor involved would render such a procedure useless for practical application. Microprocessor, with its extremely fast speed of operation, extremely low hardware requirement, flexibility of operation due to software facilities etc. is an automatic choice for most efficient implementation of test sets.

Microprocessor based IC testers may again be of two types, viz., dedicated testers and universal testers. Dedicated testers will require different hardware for different ICs and are, as such, suited for application in places where only few specific types of ICs are being used. In a laboratory already having a microprocessor based system, it will be very easy and inexpensive to design such a tester.

Universal IC tester, as the name suggests, can test different types of IC chips using the same hardware. The problems encountered in such a design are discussed in chapter 6. The hardware cost of such a tester will, definitely, be more than a dedicated IC tester, but with a common hardware, such a tester is very useful for testing a wide variety of ICs, e.g., in a store of ICs from where ICs may be distributed to different sections.

In both the IC testers, a similar testing procedure may be used. Test programmes for testing different ICs are stored in the non-erasible memory. The IC to be tested is inserted in the test socket. The programme may be executed directly by the user, after finding out the starting location of the test programme; or by pressing different keys provided on the tester in a particular sequence.

Once the test programme begins, the microprocessor continuously sends different test inputs to the chip and reads the chip outputs. These outputs are compared with the desired response through microprocessor software. In case the two do not tally, the IC is faulty. This may be indicated by switching on a RED LED or sounding a horn or displaying a fault message.

In case the two outputs tally, the microprocessor proceeds to impress subsequent test inputs sequentially till all are exhausted or a wrong output is obtained, whichever is earlier. In case all outputs tally, the IC is declared as good. This may be accomplished by switching ON a GREEN LED or by displaying a suitable message.

2.4 DEVELOPMENT OF IC TESTER

In this dissertation, a dedicated IC tester has been developed. This IC tester makes use of the HIL-2961 microprocessor trainer. Initially, various test programmes were executed through HIL-2961 keyboard. The good status of an IC is indicated by a GREEN LED and the faulty status of an IC is indicated by a RED LED accompanied by a horn. Subsequent developments, including interfacing the HIL-2961 with a CRT terminal, are discussed in later chapters.

Implementing the test programme once and obtaining satisfactory results, does not, necessarily, mean absence of intermittent faults which are time varying. For this purpose, a facility has been provided to implement the test programme repeatedly. Generally, the test programme may be repeated 1000 times or so. However, at present, this facility has been restricted to repetition of the test programme ten times. This number can be modified easily by changing the number loaded in the counter indicating number of repeat loops desired. Working on the same basis, reliability testing, using accelerated, destructive test procedure, can be incorporated. Key 5 on the HIL-2961 keyboard is programmed as the key for executing REPEAT instruction. To exit from the test programme, the user presses STOP key. Key 6 on the HIL-2961 keyboard is programmed to execute STOP instruction.

The programme for REPEAT and STOP keys was developed on the basis of keyboard management technique used by the HIL-2961 monitor. In addition, the IC tester programme uses DELB and HORN2 subroutines provided in the HIL-2961 monitor.

2.5 QUADRUPLE TWO INPUT NAND GATE IC 7400

7400 chip consists of four 2-input NAND gates. Its pin configuration is given in Appendix 'E'. The truth table for a two input NAND gate (Fig.2.1) is given in Table 2.1 for fault free operation.



| A | B | C |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 0 |
| 1 | 0 | 1 |

FIG.2.1 : TWO INPUT
NAND GATE

TABLE 2.1 : TRUTH TABLE

2.6 MINIMAL TEST SET DEVELOPMENT FOR IC 7400

The two input NAND gate shown in Fig.2.1 has three lines (two input lines and one output line). Assuming single faults leads to a total of six possible faults of the nature of s-a-1 or s-a-0.

In order to arrive at a minimal test set, we shall consider each fault separately and determine the test vector which detects this fault. The minimal combination of all such test vectors will yield the minimal test set.

For a particular fault \mathcal{L} , vector $(A,B)_i$ is a test vector iff the circuit yields different O/Ps for the circuit for healthy and faulty behaviour.

Therefore, for $(A,B)_i$ to be a test vector for fault \mathcal{L} ,

$$C_i \oplus C_i^{\mathcal{L}} = 1$$

where \oplus denotes exclusive - OR, C_i and $C_i^{\mathcal{L}}$ are O/P values of healthy and faulty gate respectively, corresponding to input vector $(A,B)_i$.

TABLE 2.2 Gives the test sets for individual faults.

| FAULT | TEST VECTOR | | C | $C^{\mathcal{L}}$ | $C \oplus C^{\mathcal{L}}$ |
|---------|-------------|---|---|-------------------|----------------------------|
| | A | B | | | |
| A s-a-1 | 0 | 1 | 1 | 0 | 1 |
| A s-a-0 | 1 | 1 | 0 | 1 | 1 |
| B s-a-1 | 1 | 0 | 1 | 0 | 1 |
| B s-a-0 | 1 | 1 | 0 | 1 | 1 |
| C s-a-1 | 1 | 1 | 0 | 1 | 1 |
| C s-a-0 | 0 | 0 | 1 | 0 | 1 |
| C s-a-0 | 0 | 1 | 1 | 0 | 1 |
| C s-a-0 | 1 | 0 | 1 | 0 | 1 |

TABLE 2.2 : TEST SETS FOR TWO INPUT NAND GATE

Complete test set is 01, 11, 10, 11, 11, 00, 01, 10. The minimal test set, therefore, is 01, 10, 11.

01 tests faults A s-a-1, C s-a-0

10 tests faults B s-a-1, C s-a-0

11 tests faults A s-a-0, B s-a-0, C s-a-1

The flow chart implementing this test set in 7400 test programme is given in Fig.2.2.

2.7 KEY DEBOUNCING

Every mechanical key is associated with a clatter whenever it is pressed or released. This results in intermittent ON and OFF signal generation for about 5 ms whenever a key is pressed or released. Typically, the signal generated whenever a particular key is pressed or released, may be as shown in Fig.2.3. In order to correctly interpret the signal from a particular key, special attention must be paid to this characteristic. In this dissertation, a software debounce programme has been developed and used for key 6 and key 5.

Having executed the test programme once, the microprocessor waits for a REPEAT or STOP instruction. It constantly scans the keyboard. Whenever it receives a signal indicating that key 6 or key 5 has been pressed, it waits for 10 ms to ignore the clatter. It then again scans the keyboard

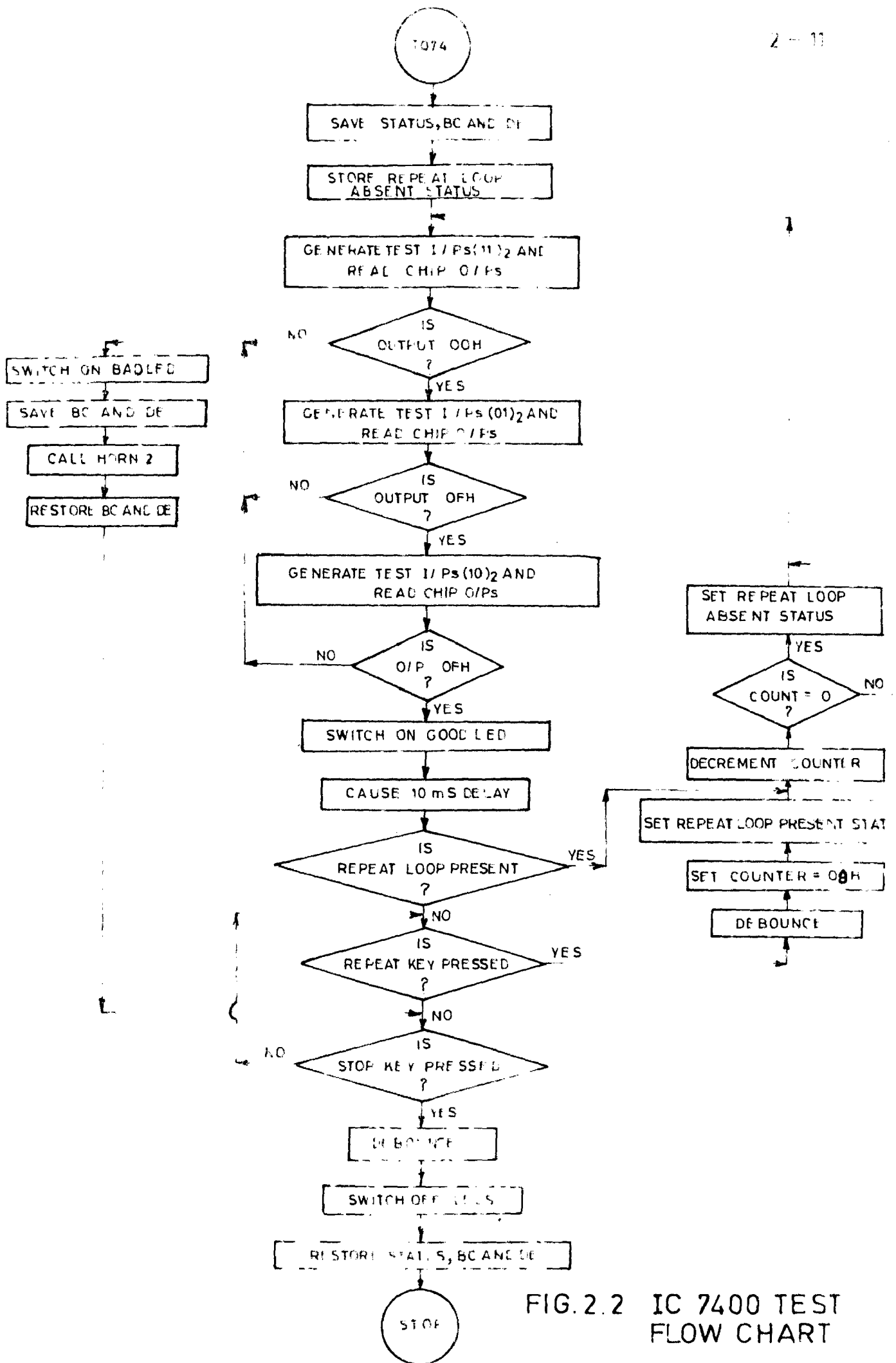


FIG. 2.2 IC 7400 TEST FLOW CHART

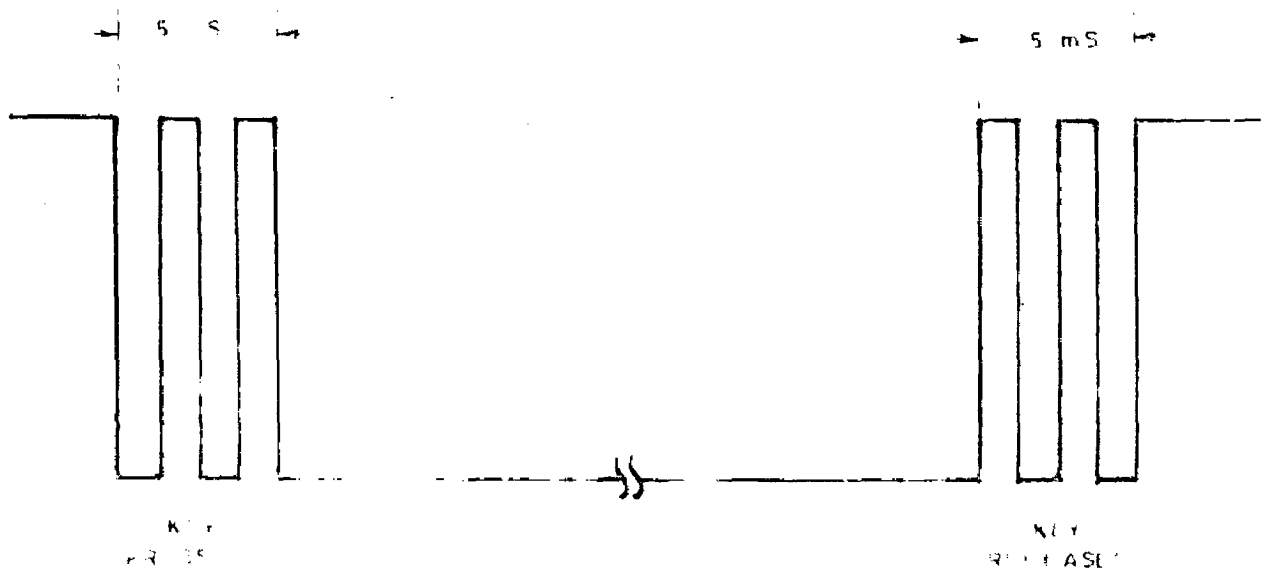


FIG.2.3 WAVEFORM DUE TO SWITCH BOUNCE

to see if the key is still pressed and waits till the operator lifts his finger from the key. On receiving the indication that the key is released, the programme again waits for 10 ms to ignore the clatter associated with key release.

2.8 PROGRAMME FOR IC 7400 TEST

The programme for IC 7400 test through HIL-2961 is labelled as 'T074' and starts from location OCBOH in the memory. Existing ports provided in HIL-2961 i.e., port 30 (O/P port) and port 20 (/P port) are used for applying test I/Ps and reading chip O/Ps. Circuit connections for 7400 test card are shown in Fig.2.4 and the PCB layout is given in Fig.2.5. The test programme is given below:

| | |
|--------------------|-----------------------|
| NAME OF SUBROUTINE | T074 |
| INPUTS | NONE |
| OUTPUTS | SIGNALS FOR LED, HORN |
| CALLS | DELB, HORN2 |
| DESTROYS | H, L, F/Fs |

DESCRIPTION T074 CHECKS UP IC 7400 USING
HIL 2961 KEYBOARD

| | | | | |
|------|--------|------|--------|--------------------------------------|
| OCBO | | T074 | | |
| OCBO | F5 | PUSH | PSW | Save status |
| OCB1 | C5 | PUSH | B | Save BC |
| OCB2 | D5 | PUSH | D | Save DE |
| OCB3 | 3E00 | MVI | A, OOH |] Store repeat loop absent status |
| OCB5 | 32FE0F | STA | RPTLP | |

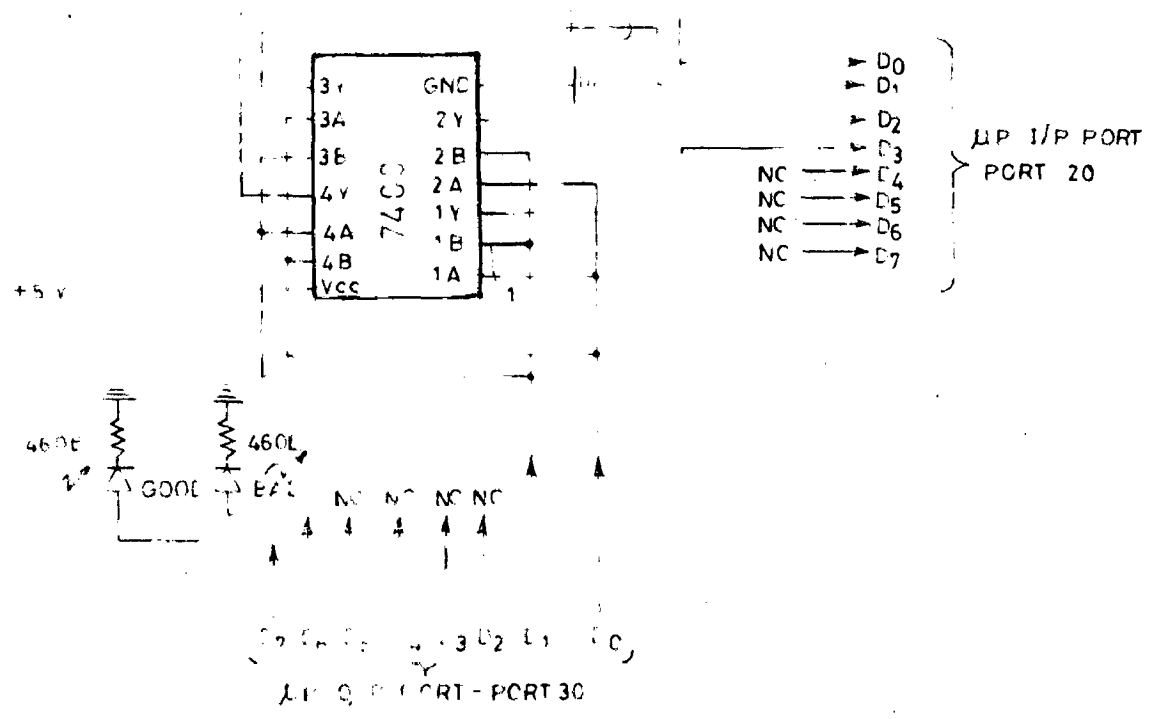


FIG. 2.4 7400 TEST CARD CIRCUIT DIAGRAM

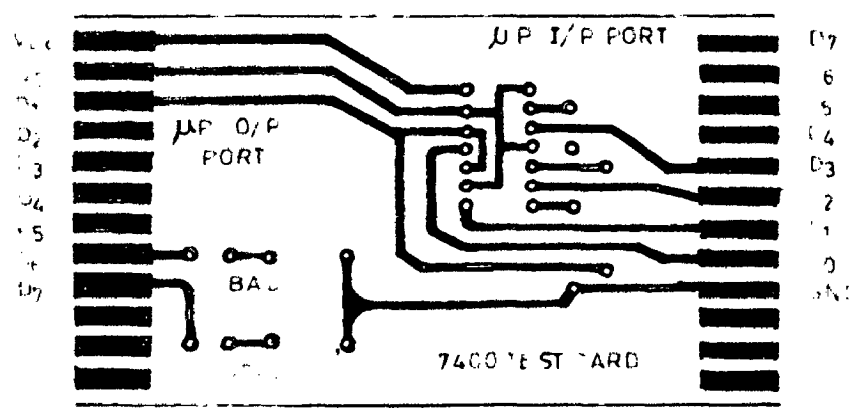


FIG. 2.5 PCB LAYOUT

| | | | | | |
|------|--------|-------|------|---------|--|
| OCB8 | | START | | | |
| OCB8 | 3E07 | | MVI | A,07H | Apply test I/Ps (11) ₂ to NAND gates |
| OCBA | D330 | | OUT | 30H | |
| OCBC | DB20 | | IN | 20H | Read chip O/Ps |
| OCBE | E60F | | ANI | 0FH | Mask unwanted bits |
| OCCO | C2270D | | JNZ | FLTND | If chip O/Ps are not correct, branch |
| OCC3 | 2E06 | | MVI | A,06H | Apply test I/Ps (10) ₂ to NAND gate |
| OCC5 | D330 | | OUT | 30H | |
| OCC7 | DB20 | | IN | 20H | Read chip O/Ps |
| OCC9 | E60F | | ANI | 0FH | Mask unwanted bits |
| OCCB | FE0F | | CPI | 0FH | If chip O/Ps are not correct, branch |
| OCCD | C2270D | | JNZ | FLTND | |
| OCDO | 3E05 | | MVI | A,05H | Apply test I/Ps (01) ₂ to NAND gate |
| OCD2 | D330 | | OUT | 30H | |
| OCD4 | DB20 | | IN | 20H | Read chip O/Ps |
| OCD6 | E60F | | ANI | 0FH | Mask unwanted bits |
| OCD8 | FE0F | | CPI | 0FH | If chip O/Ps are not correct, branch |
| OCDA | C2270D | | JNZ | FLTND | |
| OCDD | 3E80 | | MVI | A,80H | Chip is good. Switch on GOOD LED |
| OCDF | D330 | | OUT | %0H | |
| OCE1 | 010500 | | LXI | B,0005H | Cause 5 ms delay |
| OCE4 | CD3004 | | CALL | DELB | |
| OCE7 | 3AFEOF | | LDA | RPTLP | If repeat loop is present, branch |
| OCEA | FE15 | | CPI | 15H | |
| OCEC | CA570D | | JZ | RPTND | |
| OCEF | | ALPHA | | | |
| OCEF | 3EEF | | MVI | A,EFH | Set scan port to EFH |
| OCF1 | D328 | | OUT | 28H | |
| OCF3 | DB18 | | IN | 18H | Read key board I/P |
| OCF5 | E607 | | ANI | 07H | If REPEAT key (key5) is pressed, branch |
| OCF7 | FE05 | | CPI | 05H | |
| OCF9 | CA390D | | JZ | ALPHA3 | |

| | | | | | |
|-----|--------|--------|------|----------|--|
| CFC | | ALPHA1 | | | |
| CFC | 3EEF | | MVI | EFH | Erase, set scan port to EFH |
| CFE | D328 | | OUT | 28H | |
| DOO | DB18 | | IN | 18H | Read keyboard I/P |
| DO2 | E607 | | ANI | 07H | If STOP key (key6) is not pressed, try for another input |
| DO4 | FE03 | | CPI | 03H | |
| DO6 | C2EF0C | | JNZ | ALPHA | |
| DO9 | O10A00 | | LXI | B, 000AH | Cause 10 ms delay |
| DOC | CD3004 | | CALL | DELB | |
| DOF | | ALPHA2 | | | |
| OF | 3EEF | | MVI | A, EFH | Set scan port to EFH |
| D11 | D328 | | OUT | 28H | |
| D13 | DB18 | | IN | 18H | Read key board input |
| D15 | E607 | | ANI | 07H | If OFF key is pressed, wait. |
| D17 | FE03 | | CPI | 03H | |
| D19 | CAOFOD | | JZ | ALPHA2 | |
| D1C | CD3004 | | CALL | DELB | Else, wait for 10 ms |
| D1F | 3E00 | | MVI | A, 00H | Switch OFF LEDS |
| D21 | D330 | | OUT | 30H | |
| D23 | D1 | | POP | D | Restore DE |
| D24 | C1 | | POP | B | Restore BC |
| D25 | F1 | | POP | PSW | Restore status |
| D26 | 76 | | HLT | | |
| D27 | | FLTND | | | |
| D27 | 3E40 | | MVI | A, 40H | Chip is bad. Switch on 'BAD' LED |
| D29 | D330 | | OUT | 30H | |
| D2B | C5 | | PUSH | B | Save BC |
| D2C | D5 | | PUSH | D | Save DE |

| | | | | | |
|------|--------|--------|------|----------|---|
| OD2D | 0620 | | MVI | D, 20H | |
| OD2F | 1630 | | MVI | D, 30H | Sound Horn |
| OD31 | CD4704 | | CALL | HORN2 | |
| OD34 | D1 | | POP | D | Restore DE |
| OD35 | C1 | | POP | B | Restore BC |
| OD36 | C3FC0C | | JMP | ALPHA1 | Go back to programme to get next instruction. |
| OD39 | | ALPHA3 | | | |
| OD39 | 010A00 | | LXI | B, 000AH | Debounce begins. |
| OD3C | CD3004 | | CALL | DELB | Cause 10 ms delay |
| OD3F | | ALPHA4 | | | |
| OD3F | 3EEF | | MVI | A, EFH | Get an input from keyboard |
| OD41 | D328 | | OUT | 28H | |
| OD43 | DB18 | | IN | 18H | |
| OD45 | E607 | | ANI | 07H | If REPEAT key is still pressed, check again |
| OD47 | FE05 | | CPI | 05H | |
| OD49 | CA3F0D | | JZ | ALPHA4 | |
| OD4C | CD3004 | | CALL | DELB | Key released. Cause 10 ms delay. Debounce ends |
| OD4F | 110A00 | | LXI | D, 0AH | Set up repeat loop counter |
| OD52 | 3E15 | | MVI | A, 15H | Set up repeat loop present status |
| OD54 | 32EFOF | | STA | RPTLP | |
| OD57 | | RPTND | | | |
| OD57 | 7B | | MOV | A, E | Is the lower byte of counter exhausted? |
| OD58 | FE00 | | CPI | 00H | |
| OD5A | CA610D | | JZ | RPTND1 | Yes-Branch |
| OD5D | 1B | | DCX | D | No-Decrement counter |
| OD61 | | RPTND1 | | | |
| OD61 | 7A | | MOV | A, D | Lower byte of counter is zero. Is upper byte also zero? |
| OD62 | FE00 | | CPI | 00H | |

| | | | | |
|------|--------|--------|--------|-------------------------|
| OD64 | CA6B0D | JZ | RPTND2 | Yes-Branch |
| OD67 | 1B | DCX | D | No-Decrement counter |
| OD68 | C3B80C | JMP | START | Repeat programme |
| OD6B | | RPTND2 | | |
| OD6B | 3E00 | MVI | A, 00H | Counter exhausted. Load |
| OD6D | 32FE0F | STA | RPTLP | repeat loop absent |
| | | | | status |
| OD70 | C3B80C | JMP | START | Repeat programme |

PROGRAMME TABLE

| | | | | |
|------|-------|-----|-------|-------------------------------------|
| OFFE | RPTLP | EQU | OFFEH | Location for repeat loop status. |
|------|-------|-----|-------|-------------------------------------|

CHAPTER - 3INTERFACING HIL 2961 WITH CRT TERMINAL

INTRODUCTION: NEED FOR A CRT TERMINAL

As has been mentioned in Chapter 1/^{that} the visual display in the HIL 2961 is limited to six digit, seven segment LEDs. This imposes a severe handicap on the user efficiency in communicating with the trainer. For example, having entered a programme, checking for its correctness to ensure that the programme has been entered correctly, the user has to examine the address one by one. Besides being extremely slow and tedious, this method is liable to cause repeated mistakes especially when checking up a long programme. Again, while debugging a programme, the user may insert break points in the programme or execute it instruction by instruction. The results so obtained has to be noted down by the user and since the results of various instructions are not available directly, it takes very long to spot bugs in the programme.

A CRT terminal is an ideal solution to these problems. Due to the semi-permanent display provided on the CRT screen, a large amount of data is available to the user. With a suitable editor, the user can get the entire programme or large blocks of data on the screen, contents of various registers can be displayed after execution of each instruction and many such programmes can be developed to enhance user efficiency.

Interfacing a CRT terminal is, therefore, the first logical step in system development after the first stage of communicating directly with the microprocessor through a key board.

Besides, the process of developing the hardware interface and software programmes for the monitor permits the user an intimate interaction with the numerous aspects of microprocessor functioning, thereby resulting in better understanding of the microprocessor's software potential and hardware requisites.

In the context of development of IC tester, the CRT terminal can be used effectively to simplify the testing procedure. With various messages printed on the screen in response to programmed keys, any person can use the IC tester without getting involved with unnecessary details.

Obviously, it is not economically viable to produce an IC tester with a CRT terminal. However, on the basis of this work a similar approach can be used to display different messages on a smaller display using 7 segment LEDs.

Nevertheless, the IC tester developed is very useful for use in a laboratory where a CRT terminal may be already available, thereby not requiring additional investment.

3.2 INTERACTIVE DISPLAY TERMINAL ADM-3A (5)

In very general sense, CRT terminal is a device which generates unique ASCII codes in response to different keys being pressed

on its keyboard. This code is then transmitted serially to the interfaced computer. It can also accept data through the connected computer serially, decode it and display the character on the screen. For the purpose of this dissertation, an 'Interactive Display Terminal ADM-3A', marketed by Lear Siegler, Inc., was provided by the Computer Centre.

ADM-3A has a 30 cm rectangular screen which can display 24 lines each having a maximum of 80 characters. Its keyboard layout is identical to that of a standard type-writer and has 59 keys. It can communicate over a large range of baud rates from 75 to 19200, switch selectable. It permits both RS-232C(6,7) and 20 mA current loop interface and has many other features described in Appendix 'D'.

For interfacing ADM-3A with HIL-2961 following features were selected.

1. RS-232C interface
2. Full Duplex mode
3. 300 baud rate
4. 7 bit data word length
5. Bit 8 forced to zero
6. Parity inhibited
7. 2 stop bits
8. AUTO NL mode
9. Upper case alphabetic characters' option.

3.3 DEVELOPMENT OF INTERFACE

Development of an interface between the CRT terminal and the microcomputer has the following two main aspects.

1. Hardware interface: It must satisfy the following requirements:
 - (a) Change TTL level signals from the microcomputer to RS-232C level signals for the CRT and vice versa,
 - (b) Convert parallel data from the microcomputer into serial form for transmission to the CRT terminal, and similarly convert serial data from the CRT into parallel 8-bit characters for the microcomputer; and
 - (c) To attach relevant framing information to each character received from the microcomputer for transmission to the CRT terminal and to remove this information before transmitting a character from CRT terminal to the microcomputer.

It may be noted that the hardware interface may be considerably simplified by using the SID and SOD lines of the 8085A. The microcomputer can then communicate serially with the CRT terminal without necessitating a serial to parallel conversion and vice-versa. In case of HIL 2961, the SOD line is connected permanently to a speaker and, therefore, not available unless this connection is cut. Another factor against the use of

the SID and SCD lines is the time wasted by the microcomputer in sending and receiving data serially on these lines. This implies underutilization of the microprocessor efficiency.

2. Software interface: The functions outlined in 1(b) and (c) are fulfilled by the 8251A USART. The 8251A requires a software programme to enable its function in the desired manner. The software interface is also required for programming different keys on the CRT terminal keyboard for executing different commands on the microcomputer through the CRT terminal.

The hardware interface is akin to a handshake between two strangers and the software interface is like a common language the two must speak for an intelligent information exchange.

3.4 HARDWARE INTERFACE

The hardware interface development consisted of developing following units:

1. +5V, +12V and -10V power supply.
2. Baud Rate Generator
3. RS-232C and 8251A USART interface

Each of these will be discussed in subsequent sections.

3.5 POWER SUPPLY

The power supply is used for providing +5V supply to different ICs and +12V and -12V supplies are used for different transistors used in the RS-232C/^{interface.} Since the hardware circuitry is designed to operate correctly at the specified voltages, there is an obvious need for a regulated power supply. Regulated +12V and -10V supplies were obtained using ICs 723 and 741. The +5V regulated supply was obtained using IC 7805. The circuit diagram for the power supply is shown in Fig.3.1 and the PCB layout is shown in Fig.3.2.

3.6 BAUD RATE GENERATOR

Using the 2MHz clock available at CLK OUT terminal (Pin 37) of the 8085A CPU (available at Pin 28 of the extension connector of HIL 2961), a variable baud rate generator has been developed. This can provide switch selectable baud rates of 19200, 9600, 4800, 2400, 1760 and 1200. When used in conjunction with the 'baud rate factor' facility provided in the INTEL 8251A USART/^{it} can enable the user to operate the CRT terminal at different baud rates. In addition, using the 1760 baud rate and a baud rate factor of 16 in the 8251A, an interface for interfacing the HIL 2961 with a tele-typewriter can also be developed.

IC 74161 is the basic chip used for generating different baud rates. It is a synchronous, presetable, 4 bit binary counter. The synchronous feature is useful for high frequency

operation and the pre-settable feature reduces the external hardware to a bare minimum. The pin configuration and other details of 74161 are given in Appendix 'E'.

3.6.1 Design Calculations

Original frequency : 2MHz

Frequencies desired : 19200, 9600, 4800, 2400,
1760, 1200

Dividing factors : As per table 3.1

TABLE 3.1 : Dividing Factor Calculations

| <u>Frequencies required.</u> | <u>Dividing factors</u> |
|------------------------------|-------------------------|
| 19200 | 13,8 |
| 9600 | 13,16 |
| 4800 | 13,16,2 |
| 2400 | 13,16,4 |
| 1760 | 14,16,5 |
| 1200 | 13,16,8 |

Three 74161s are used. The first one is used as $\div 13$ or $\div 14$ counter (first state), the second one is used as $\div 2$, $\div 4$, $\div 5$ or $\div 8$ counter (third stage), the third one is used as $\div 9$ or $\div 16$ counter (second stage). $\div 13$, $\div 14$ and $\div 5$ counts are obtained by programming the counter. Appropriate inputs are placed at terminals 3,4,5 and 6, and the counter begins counting sequence from states determined by the particular combination of inputs.

3.5 POWER SUPPLY

The power supply is used for providing +5V supply to different ICs and +12V and -12V supplies are used for different transistors used in the RS-232C/ interface. Since the hardware circuitry is designed to operate correctly at the specified voltages, there is an obvious need for a regulated power supply. Regulated +12V and -10V supplies were obtained using ICs 723 and 741. The +5V regulated supply was obtained using IC 7805. The circuit diagram for the power supply is shown in Fig.3.1 and the PCB layout is shown in Fig.3.2.

3.6 BAUD RATE GENERATOR

Using the 2MHz clock available at CLK OUT terminal (Pin 37) of the 8085A CPU (available at Pin 28 of the extension connector of HIL 2961), a variable baud rate generator has been developed. This can provide switch selectable baud rates of 19200, 9600, 4800, 2400, 1760 and 1200. When used in conjunction with the 'baud rate factor' facility provided in the INTEL 8251A USART/^{it} can enable the user to operate the CRT terminal at different baud rates. In addition, using the 1760 baud rate and a baud rate factor of 16 in the 8251A, an interface for interfacing the HIL 2961 with a tele-typewriter can also be developed.

IC 74161 is the basic chip used for generating different baud rates. It is a synchronous, presetable, 4 bit binary counter. The synchronous feature is useful for high frequency

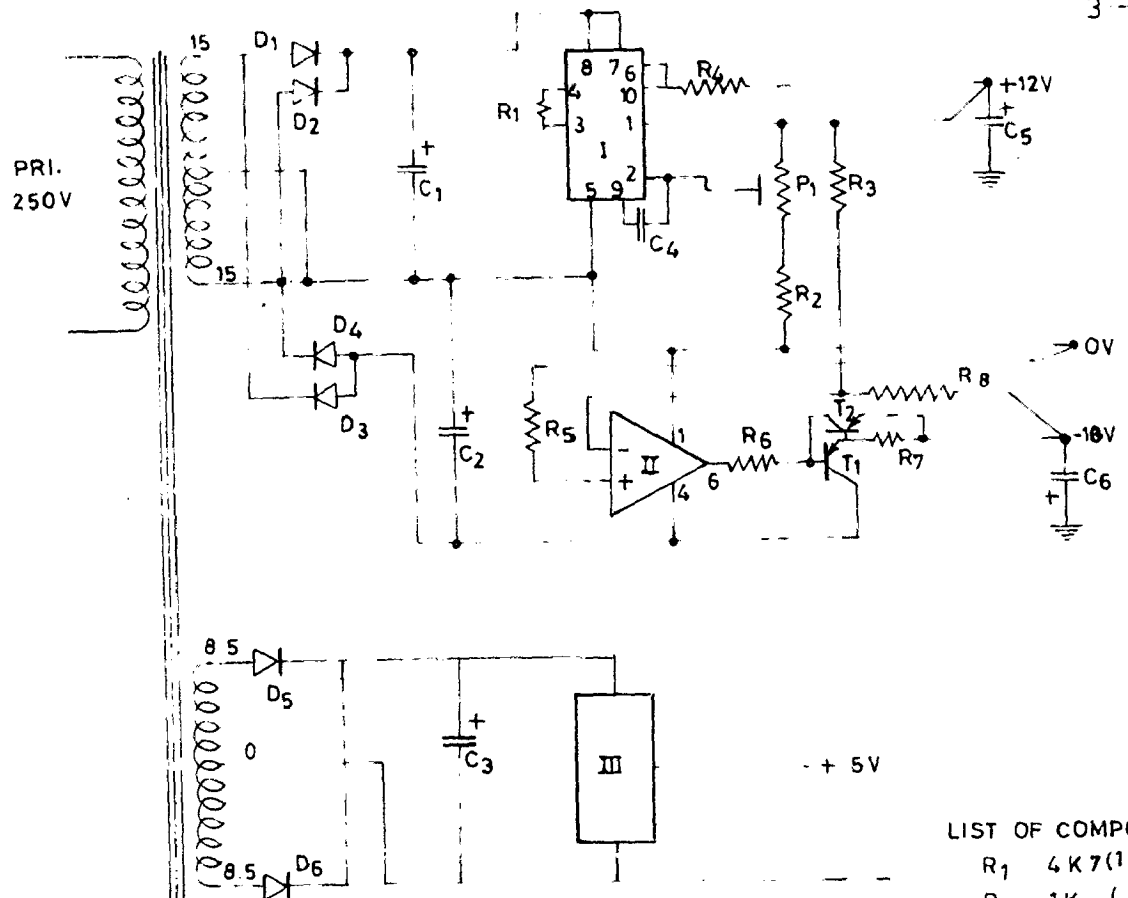


FIG. 3.1 CIRCUIT DIAGRAM FOR REGULATED POWER SUPPLY

LIST OF COMPONENTS

- R₁ 4K7 (1/4W)
- R₂ 1K (»)
- R₃ 12K (»)
- R₄ 56E (1W)
- R₅ 1K (1/4W)
- R₆ 1K5 (»)
- R₇ 56E (1W)
- R₈ 10K (1/4W)
- C₁ 500,25
- C₂ 500,25
- C₃ 1000,25
- C₄ 100PF
- C₅ 10,12
- C₆ 10,12
- D₁ D₆ 1N4001
- IC1 723
- IC2 741
- IC3 7805
- P₁ 10K PRESET
- T₁ 2x SK100
- T₂ BC158

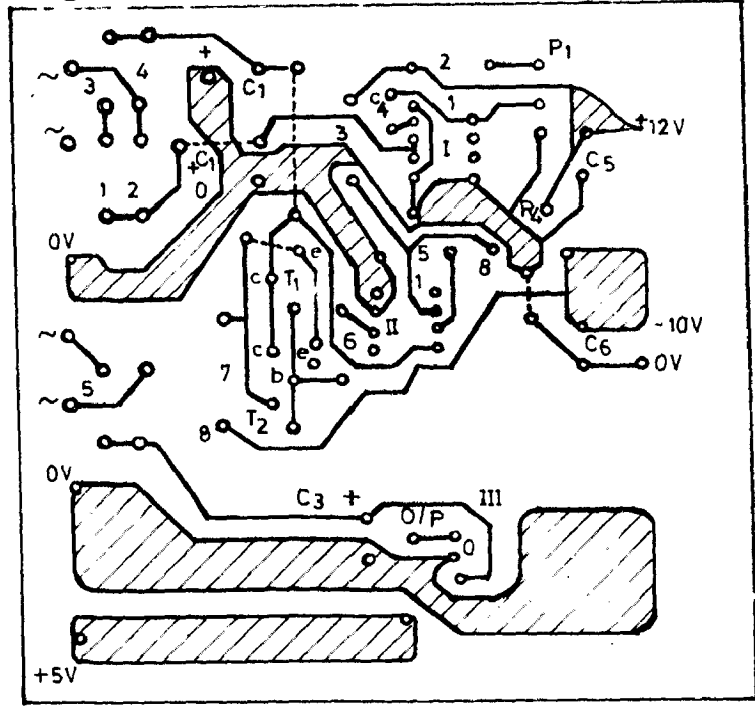


FIG. 3.2 PCB LAYOUT

operation and the pre-settable feature reduces the external hardware to a bare minimum. The pin configuration and other details of 74161 are given in Appendix 'E'.

3.6.1 Design Calculations

Original frequency : 2MHz

Frequencies desired : 19200, 9600, 4800, 2400,
1760, 1200

Dividing factors : As per table 3.1

TABLE 3.1 : Dividing Factor Calculations

| <u>Frequencies required.</u> | <u>Dividing factors</u> |
|------------------------------|-------------------------|
| 19200 | 13,8 |
| 9600 | 13,16 |
| 4800 | 13,16,2 |
| 2400 | 13,16,4 |
| 1760 | 14,16,5 |
| 1200 | 13,16,8 |

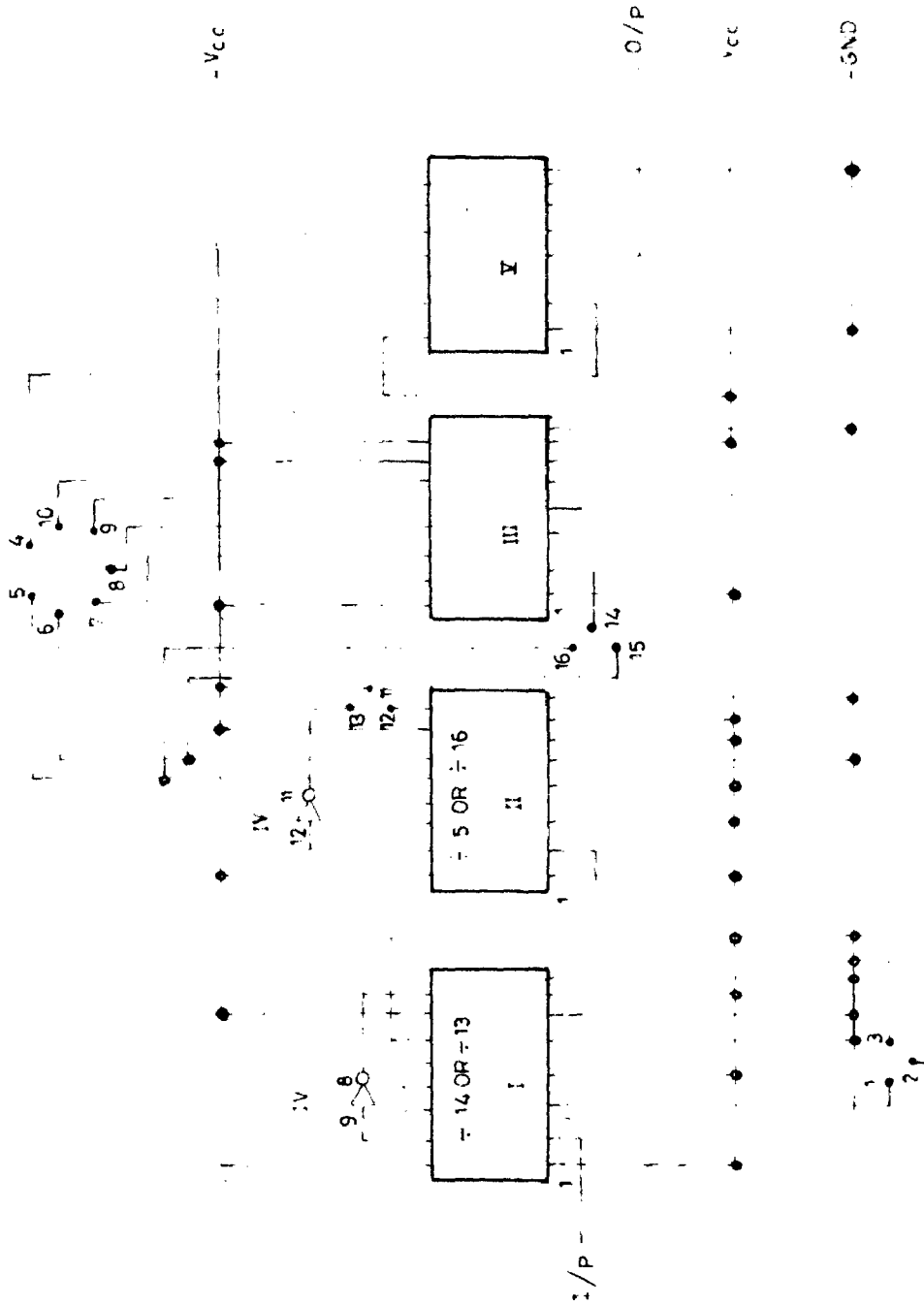
Three 74161s are used. The first one is used as $\div 13$ or $\div 14$ counter (first stage), the second one is used as $\div 2$, $\div 4$, $\div 5$ or $\div 8$ counter (third stage), the third one is used as $\div 9$ or $\div 16$ counter (second stage). $\div 13$, $\div 14$ and $\div 5$ counts are obtained by programming the counter. Appropriate inputs are placed at terminals 3,4,5 and 6, and the counter begins counting sequence from states determined by the particular combination of inputs.

The circuit diagram for the baud rate generator PCB is shown in Fig.3.3. In order to provide a buffered output, a 74125 buffer has been used. The PCB layout is shown in Fig.3.4. In order to make the baud rate generator an independent unit, separate power supply has been provided for it. The regulated +5V supply uses a 7805IC.

3.6.2 Operating the Baud Rate Generator

The front panel of the baud rate generator has two select switches. Right hand side selector switch labelled 'MASTER SELECT' selects either 1760 or the group of other baud rates. For individual baud rates, the left select switch labelled 'RATE SELECT' is then used. The recommended operating procedure is as under:

1. Before connecting the unit to power supply, ensure that both the select switches are in OFF position and the power on switch is OFF.
2. Select appropriate switch positions for the desired baud rate.
3. Connect the CLK OUT terminal of 8085A to the terminal marked I/P on the front panel. Connect the Ground terminal to terminal marked GND.



COMPONENTS
 I, II, III, IV, V : 74161

| BAUD RATES | WIRE CONNECTIONS |
|------------|---|
| 19200 | 1, 2, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100 |
| 9600 | 1, 2, 12, 10, 11, 14, 10, 15, 14, 10, 6 |
| 4800 | 1, 2, 12, 10, 11, 14, 10, 15, 14, 10, 6 |
| 2400 | 1, 2, 12, 10, 11, 14, 10, 15, 14, 10, 6 |
| 1200 | 1, 2, 12, 10, 11, 14, 10, 15, 14, 10, 6 |
| 600 | 1, 2, 12, 10, 11, 14, 10, 15, 14, 10, 6 |

FIG. 3.3 BAUD RATE GENERATOR
 CIRCUIT DIAGRAM

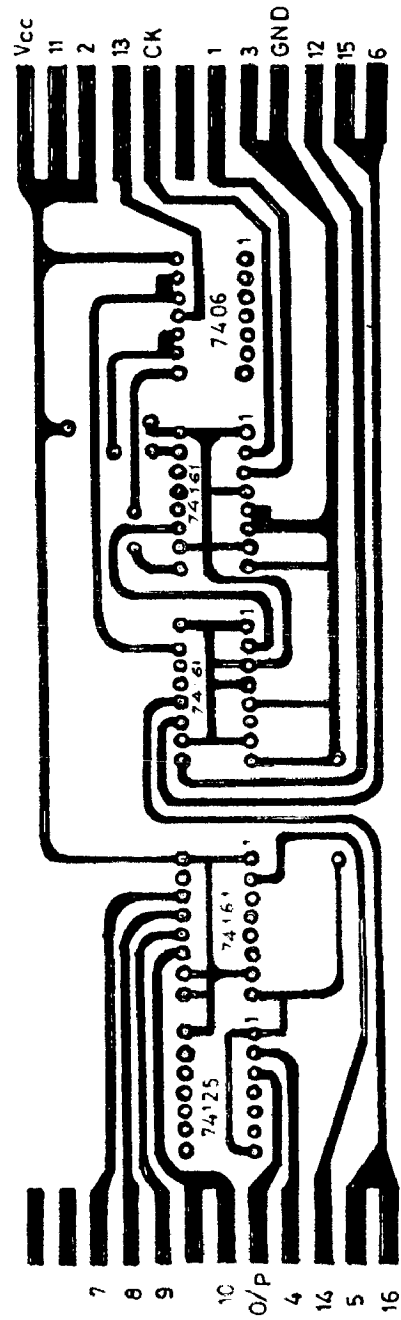


FIG. 3.4 PCB LAYOUT

4. Connect 230V supply and switch ON the power ON switch.
5. Switch on the trainer to activate the 2MHz clock.
6. The desired baud rate is available at terminal marked O/P.

3.7 RS-232C AND 8251A USART INTERFACE (8)

ADM-3A offers two possible interfaces viz., RS-232C and 20 mA current loop. In this dissertation RS-232C option was chosen. RS-232C uses negative true logic for its operation. -15V to -5V is recognised as logic 1 and +5V to +15V is recognised as logic 0 level. TTL levels used by HIL 2961, on the other hand, use positive true logic, recognising 2.4 V to 5.25V as logic 1 and 0 V to .8V as logic 0 (9).

'RS 232C and 8251A USART interface' is the unit which performs the functions of a hardware interface outlined in section 3.3. Power supply and the baud rate generator are used as supporting units to make this unit functional. The circuit diagram of this interface is shown in Fig.3.5 and the PCB layout is given in Fig.3.6.

The ADM-3A terminal operates in asynchronous format, the 8251A can operate both in the synchronous and the asynchronous format. In order to understand the microcomputer to CRT terminal interface, it is necessary to understand the techniques of serial data transmission. Appendix 'F' provides a brief description of communication formats with special emphasis on asynchronous format.

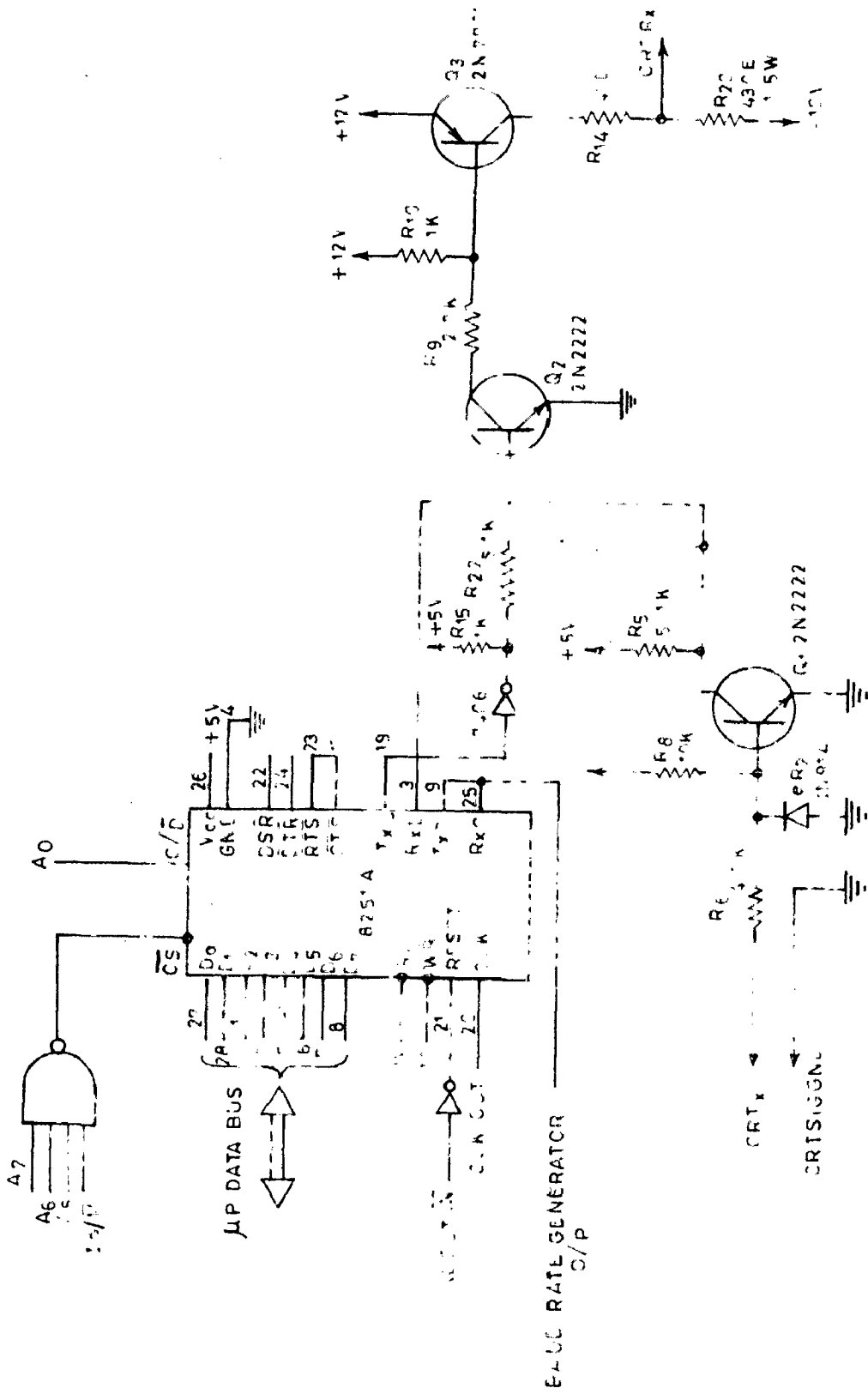


FIG. 3.5 RS 232 C AND 8251 A USART INTERFACE

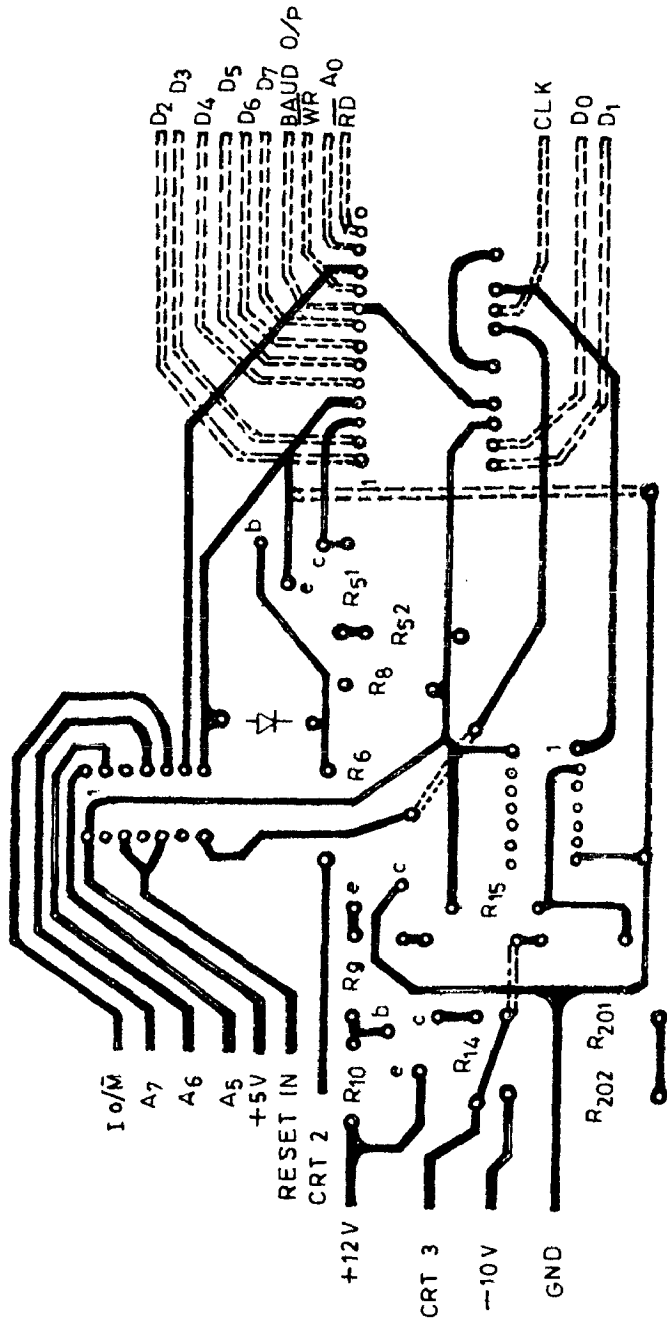


FIG. 3.6 RS 232 C AND 8251 A
US ART INTERFACE PCB LAYOUT

3.7.1 INTEL 8251A USART

INTEL's 8251A 'Universal Synchronous Asynchronous Receiver Transmitter' is an IC chip that facilitates data transfer between a parallel and serial device. The functions of a USART are as follows:

1. Receive parallel data from the microprocessor, convert it into serial form and encode it as per the required format.
2. Transmit this data to the I/O device.
3. Receive serial data from the I/O device, decode it, convert it into parallel data and send it to the microprocessor.
4. Generate appropriate control signals to inform the microprocessor that it is ready to receive a word from it or that it already has a word yet to be transmitted to the I/O device, to inform the microprocessor that it has a data word for it from the I/O device. It also checks the data word for correct format and raises appropriate signals for correctness/fault.

The detailed description of 8251A is given in Appendix 'G'. The operation of the 8251A in the interface developed is as under. Pins D_0 - D_7 (Pin Nos 27,28,1,2,5,6,7,8) are connected to the microprocessor data bus. TXD (pin 19) is

used for either loading the mode and control instruction, or for reading the Status Register. The $\overline{\text{RTS}}$ (Pin 23) and the $\overline{\text{CTS}}$ (Pin 17) terminals are shorted together to enable data transmission. Other pins of the 8251A are not used.

Once all external connections are made with the interface unit, and the power is switched ON on all supporting units, the interface becomes functional. However, before the interface can commence operation, the 8251A must be programmed. This will be discussed in the software development described in Chapter 4.

3.7.1 INTEL 8251A USART

INTEL's 8251A 'Universal Synchronous Asynchronous Receiver Transmitter' is an IC chip that facilitates data transfer between a parallel and serial device. The functions of a USART are as follows:

1. Receive parallel data from the microprocessor, convert it into serial form and encode it as per the required format.
2. Transmit this data to the I/O device.
3. Receive serial data from the I/O device, decode it, convert it into parallel data and send it to the microprocessor.
4. Generate appropriate control signals to inform the microprocessor that it is ready to receive a word from it or that it already has a word yet to be transmitted to the I/O device, to inform the microprocessor that it has a data word for it from the I/O device. It also checks the data word for correct format and raises appropriate signals for correctness/fault.

The detailed description of 8251A is given in Appendix 'G'. The operation of the 8251A in the interface developed is as under. Pins D_0 - D_7 (Pin Nos 27,28,1,2,5,6,7,8) are connected to the microprocessor data bus. TXD (pin 19) is

connected to 'Receive Data' pin (pin 3) of ADM-3A. RXD (pin 3) is connected to 'Transmit Data' pin (pin 2) of the ADM-3A. Connections from 8251A to ADM-3A carry the necessary circuitry for changing the voltage levels appropriately to TTL level or RS-232C. TXC (Pin 9) and RXC (Pin 25) are shorted and connected to the output of the band rate generator. CLK (Pin 20) is connected to CLK OUT terminal of 8085A which is of 2MHz frequency. \overline{RD} , \overline{WR} , V_{CC} and GND are appropriately connected. RESET (Pin 21) is connected to the $\overline{RESET\ IN}$ terminal of 8085A through an inverter. In the present design, the 8251A can be addressed by any port number between E0 to FF. Accordingly, the \overline{CS} (Pin 11) terminal is connected to the output of a four input NAND gate having 8085 signals A_7 , A_6 , A_5 and IO/M as its inputs. The other select terminal C/\overline{D} (Pin 12) is connected directly to address line A_0 of the microprocessor. In this dissertation, the 8251A is addressed either as E0 or E1. Addressing it as port E0 enables data transfer between USART and the microprocessor, while the E1 address is

used for either loading the mode and control instruction, or for reading the Status Register. The $\overline{\text{RTS}}$ (Pin 23) and the $\overline{\text{CTS}}$ (Pin 17) terminals are shorted together to enable data transmission. Other pins of the 8251A are not used.

Once all external connections are made with the interface unit, and the power is switched ON on all supporting units, the interface becomes functional. However, before the interface can commence operation, the 8251A must be programmed. This will be discussed in the software development described in Chapter 4.

CHAPTER - 4MONITOR FOR THE CRT TERMINAL (10)

4.1 INTRODUCTION

As discussed in Section 3.3, a software interface is necessary for intelligent interchange of information between the microprocessor and the CRT terminal. This software performs two distinct functions:

1. Programming the 8251A : For proper working, the USART must be programmed correctly before commencing data transfer. This involves loading the appropriate mode word, and control word into the 8251A control register.
2. Programming ADM-3A keys for specific tasks: This involves developing programmes for identifying different keys and executing commands represented by them.

Accordingly, different programmes have been developed. All these programmes together constitute the monitor for the IC tester. Different programmes in the monitor developed in this dissertation shall be discussed in subsequent sections. The complete monitor listing is available at the end of this Chapter.

4.2 SIGN ON MESSAGE PROGRAMME

Once the hardware interface has been activated, the execution of this programme results in the initial sign on message, 'IC. TESTER ■ READY' to be displayed on the ADM-3A screen. The Control word used is CEH, i.e., $(11001110)_2$. This implies that the data communication is in asynchronous format, Character length is 8 bits, 2STOP bits will be added to each character, parity is not being used and the baud rate factor is 1/16. Since the ADM-3A baud rate has been selected as 300, therefore the baud rate generator output, connected to TxC & RxC terminals of 8251A, must be selected as 4800 baud. The mode word used is $(27)_H$. This implies that RTS will be held LOW, Receive and Transmit functions are enabled and the DTR (Data Terminal Ready) signal is enabled. Since RTS is connected directly to CTS, CTS will also be forced LOW and this will enable data transmission. This programme labelled 'SIGNON' resides in memory location 0800H to 0813H in the monitor.

4.3 GET COMMAND PROGRAMME

After displaying the initial sign - on message, the monitor requests the user for a command by displaying a prompt character "." This involves polled data transfer. The status register of the 8251A is repeatedly checked to see if any character has been received by the receiver, i.e., to check if a key has been pressed on the key board. Once a key

is pressed on the key board, the programme compares the code generated against the list of valid commands for the monitor. In case the code signifies one of the valid commands, the control is passed to the concerned programme for further execution. Otherwise, an asterik (*) signifying erroneous command is displayed on the terminal screen and the monitor once again waits for a valid command from the user. This programme is labelled 'GETCMD' and resides in memory locations 0816H to 083DH in the monitor.

4.4 FUNCTIONAL COMMANDS

The programmes discussed in Section 4.2 and 4.3 are automatically executed when the control is transferred to the ADM-3A keyboard from the HIL 2961 keyboard by executing the programme from 0800H. After the control is transferred to the ADM-3A keyboard, the HIL 2961 can regain the control at any time by pressing the RESET key on HIL-2961 keyboard.

Three functional commands have been provided in the IC tester monitor :

1. TRY command
2. LIST command
3. EXECUTE command

Function of these commands shall be explained individually later in this chapter. The monitor has been so programmed that each command must be terminated by a CARRIAGE RETURN.

Any other character used for terminating the command shall make the command invalid. This gives the user the option of abandoning a command at any stage before the RETURN key is pressed.

4.5 TRY COMMAND

This command is represented by key T on the ADM-3A key board. On pressing T, the monitor responds by displaying 'ICNO:'. The user should now type the number of the IC. The numbers will be displayed on screen. The monitor recognises four hexadecimal digits as valid number of digits. If less than four digits have been typed, the monitor will wait for the remaining digits and initiate no other action. If more than four digits are typed, the monitor will cause the error character to be displayed and revert to GET CMD programme. The monitor recognises T command as a valid command only for the 74 series of TTL. Typing any other number in the first two digits results in error character display. Every time an error character is displayed, the monitor jumps to the GET CMD programme, abandoning whichever command it was receiving. With small changes, T command can be used for other IC series also. Once all the four digits have been typed, the command must be terminated by the RETURN key. In case the IC tester has a software programme for testing the IC mentioned, it sends the following message:

AVAILABLE

LOCATION :

The memory location, at which the programme for testing the IC in question begins, shall be displayed in front of LOCATION:

In case the IC tester does not have a software programme for testing the IC, following message will be displayed:

NOT AVAILABLE

In both the cases, after displaying the message, the monitor shall revert to the GET CMD programme. The TRY command programme is labelled as TRY and resides in memory locations 088AH to 089AH in the monitor.

4.6 LIST COMMAND

This command is represented by key L on the ADM-3A key. Like other commands, this command also must be terminated by RETURN. The complete command is as follows :

L RET

This causes the monitor to list all the ICs for which a diagnostic programme is available. List command is labelled LIST and resides in memory location 0947H to 095DH in the monitor.

4.7 EXECUTE (GO) COMMAND

This command is represented by key G on the ADM-3A key board. A valid GO command consists of typing G followed by four valid hexa-decimal digits and terminated by the RETURN key.

In case more than four hexa decimal digits are typed, the monitor recognises only the last four digits as the starting address of the programme to be executed. Once a valid command is received, the address typed in by the user is loaded in the programme counter of the μ P and control is transferred to the programme. After execution of the programme, the control is transferred back to the monitor which goes back to the 'GETCMD' programme. The programme for EXECUTE command is labelled as GCMD and it resides in memory location 095EH to 097CH in the monitor.

4.8 SUB-ROUTINES FOR PROGRAMMES

Many sub-routines like CI, CO, ERROR, GETHX etc. are frequently called by various programmes listed in previous sections. These programmes can be understood by studying the comment field provided along with the programme in the monitor listing.

4.9 MONITOR LISTING

Complete monitor listing is given in Table 4.1. First column gives the first memory location in hexa-decimal code for the corresponding instruction, the second one gives the content(s) of memory location (s) for the instruction in hexa-decimal, the third column carries the label accorded to the instruction, if any, fourth column specifies the mnemonic for the instructions. Operands taking part in the instruction are specified in the fifth column and the sixth column represents the comment field. A brief description of the programme is given at the beginning of each programme.

TABLE 4.1 : MONITOR LISTING

SIGNON PROGRAMME

NAME SIGNON
 INPUTS NONE
 OUTPUTS SIGN ON MESSAGE 'I.C. TESTER ■ READY'
 DISPLAY ON SCREEN

CALLS CO
 DESTROYS A,B,C,E,L,F/Fs

DESCRIPTION SIGN ON message is displayed on screen. The USART is assumed to come up in RESET position (This will be taken care of by the hardware). Once the USART has been initialized, subsequent entry for printing the SIGN ON message and for transfer control to ADM-3A from HIL 2961 keyboard must be made at location 0808H.

| | | | | |
|------|--------|------|--------|--|
| 0800 | SIGNON | | | |
| 0800 | 3E2E | MVI | A,MODE | Output mode word to |
| 0802 | D3E1 | OUT | CNTL | USART control port. |
| 0804 | 3E27 | MVI | A,CMD | Output command word to |
| 0806 | D3E1 | OUT | CNTL | USART Control port. |
| 0808 | SIGN1 | | | Restart point |
| 0808 | 21000F | LXI | H,MSG | Load memory pointer with address of SIGN ON message |
| 080B | 0614 | MVI | B,LMSG | Load number of character in SIGN ON message in B |
| 80D | SIGN05 | | | |
| 80D | 4E | MOV | C,M | Get the character in C. |
| 80E | CD6408 | CALL | CO | Output character to screen |
| 811 | 23 | INX | H | Move memory pointer to next character |
| 812 | 05 | DCR | B | Reduce number of characters to be output by 1. |
| 813 | C20D08 | JNZ | SIGN05 | Repeat if all characters not displayed. |

GET COMMAND PROGRAMME

NAME GETCMD
 INPUTS NONE
 OUTPUTS PROMPT CHARACTER '.' DISPLAYED ON SCREEN

CALLS ECHO, CI, ERROR
 DESTROYS A, B, C, H, L, F/Fs

DESCRIPTION GETCMD PLACES A PROMPT CHARACTER ON THE SCREEN AND WAITS TO RECEIVE AN INPUT CHARACTER FROM THE USER. IT THEN ATTEMPTS TO LOCATE THIS CHARACTER IN ITS COMMAND CHARACTER TABLE. IF SUCCESSFUL, THE ROUTINE CORRESPONDING TO THIS CHARACTER IS SELECTED FROM A TABLE OF FUNCTIONAL COMMAND ADDRESSES, AND CONTROL IS TRANSFERRED TO THIS ROUTINE. OTHERWISE, CONTROL IS TRANSFERRED TO ERROR ROUTINE

| | | | | |
|------|--------|------|----------|---|
| 0816 | GETCMD | MVI | | |
| 0816 | 0E2E | MVI | C, '.' | Send prompt character to C |
| 0818 | CD3E08 | CALL | ECHO | Send prompt character to display |
| 081B | CD5708 | CALL | CI | Get character from user |
| 081E | CD3E08 | CALL | ECHO | Display it on the screen |
| 0821 | 79 | MOV | A, C | Get character to A |
| 0822 | 010300 | LXI | B, NCMDS | Load BC pair with number of valid functional commands |
| 0825 | 21140F | LXI | H, CTAB | Load memory pointer with address of valid commands' table |
| 0828 | GET05 | | | |
| 0828 | BE | CMP | M | Compare table entry and character |
| 0829 | CA3408 | JZ | GET10 | Branch if equal-command recognized |
| 082C | 23 | INX | H | Else, increment table pointer. |

| | | | | |
|------|--------|------|---------|---|
| 082D | 0D | DCR | C | Decrement number of commands to be compared. |
| 082E | C22808 | JNZ | GET05 | Branch if not at table end |
| 0831 | C36F08 | JMP | ERROR | Else, command character is illegal. |
| 0834 | GET10 | | | |
| 0834 | 21170F | LxI | H,CADDR | Valid command, load address of table of functional command addresses. |
| 0837 | 09 | DAD | B | Add what is left of number of valid commands |
| 0838 | 09 | DAD | B | Add again each entry in CADDR is 2 bytes long |
| 0839 | 7E | MOV | A,M | Get lower byte of address of table entry to A. |
| 083A | 23 | INX | H | Point to next byte in table |
| 083B | 66 | MOV | H,M | Get upper byte address in H |
| 083C | 6F | MOV | L,A | Move lower byte address to L |
| 083D | E9 | PCHL | | Transfer control to functional command programme |

NEME OF SUBROUTINE ECHO

INPUTS C- CHARACTER TO ECHO TO TERMINAL

OUTPUT C- CHARACTER ECHOED TO TERMINAL

CALLS CO

DESTROYS A,B,F/Fs

DESCRIPTION CHARACTER CONTAINED IN C IS DISPLAYED ON TERMINAL. A CARRIAGE RETURN IS ECHOED AS A CARRIAGE RETURN LINE FEED, AND AN ESCAPE CHARACTER IS ECHOED AS \$.

| | | | | | |
|------|--------|--------|--------|--|--|
| 083E | | ECHO | | | |
| 083E | 41 | MOV | B,C | Save argument | |
| 083F | 3E1B | MVI | ESC |] See if Echoing an Escape character | |
| 0841 | B8 | CMP | B | | |
| 0842 | C24708 | JNZ | ECHO05 | No-Branch | |
| 0845 | 0E24 | MVI | C,\$ | Yes, Echo as \$ | |
| 0847 | | ECHO05 | | | |
| 0847 | CD6408 | CALL | CO | Output through Monitor | |
| 084A | 3E0D | MVI | CR |] See if character echoed was a CARRIAGE RETURN | |
| 084C | B8 | CMP | B | | |
| 084D | C25508 | JNZ | ECHO10 | No-No need for special action | |
| 0850 | 0E0A | MVI | C,LF |] Yes -- Echo line feed also | |
| 0852 | CD6408 | CALL | CO | | |
| 0855 | | ECHO10 | | | |
| 0855 | 48 | MOV | C,B | Restore argument | |
| 0856 | 09 | RET | | | |

NAME OF SUBROUTINE CI
INPUTS NONE
OUTPUTS A AND C -- CHARACTER FROM CONSOLE
CALLS NOTHING
DESTROYS A,C,F/Ps

DESCRIPTION CI WAITS UNTIL A CHARACTER HAS BEEN
ENTERED AT THE CONSOLE. IT THEN RETURNS
THE CHARACTER IN A AND C REGISTERS.

| | | | | |
|------|--------|-----|------|------------------------------------|
| 0857 | | CI | | |
| 0857 | DBE1 | IN | CNTL | Get USART status |
| 0859 | E602 | ANI | 02H | Check for receiver buffer ready |
| 085B | CA5708 | JZ | CI | Not yet - Wait |

| | | | | |
|------|------|-----|-------|---------------------------------------|
| 085E | DBE0 | IN | DATA | Yes - Get character to A |
| 0860 | E67F | ANI | PRTYO | Turn off party bit, if set by console |
| 0862 | 4F | MOV | C,A | Put value in C |
| 0863 | C9 | RET | | |

NAME OF SUBROUTINE CO
 INPUTS C- CHARACTER TO CONSOLE
 OUTPUTS C- CHARACTER OUTPUT TO CONSOLE
 CALLS NOTHING
 DESTROYS A,F/Fs

DESCRIPTION CO WAITS UNTIL THE CONSOLE IS READY TO ACCEPT A CHARACTER AND THEN SENDS INPUT ARGUMENT TO CONSOLE

| | | | | |
|------|--------|-----|-------|---------------------------|
| 0864 | CO | | | |
| 0864 | DBE1 | IN | CNTRL | Get USART status |
| 0866 | E601 | ANI | 01H | See if transmitter ready |
| 0868 | CA6408 | JZ | CO | No - Wait |
| 086B | 79 | MOV | A,C | Yes - Move character to A |
| 086C | D3E0 | OUT | DATA | Send to console |
| 086E | C9 | RET | | |

NAME OF SUBROUTINE ERROR
 INPUTS NONE
 OUTPUTS ERROR CHARACTER TO CONSOLE
 CALLS ECHO, CROUT, GETCMD
 DESTROYS A,B,C,F/Fs

DESCRIPTION PRINTS ERROR CHARACTER (ASTERIK) ON THE SCREEN, FOLLOWED BY A CARRIAGE RETURN, LINE FEED AND THEN RETURNS CONTROL TO GETCMD.

| | | | | |
|------|--------|------|--------|---|
| 086F | ERROR | | | |
| 086F | OE2A | MVI | C,* |] Send * to console |
| 0871 | CD3E08 | CALL | ECHO | |
| 0874 | CD7A08 | CALL | CROUT | Send line feed, carriage return to console |
| 0877 | C31608 | JMP | GETCMD | Try again for another command |

| | |
|--------------------|------------|
| NAME OF SUBROUTINE | CROUT |
| INPUTS | NONE |
| OUTPUTS | NONE |
| CALLS | ECHO |
| DESTROYS | A,B,C,F/Fs |

DESCRIPTION CROUT SENDS A CARRIAGE RETURN (AND HENCE A LINE FEED) TO THE CONSOLE.

| | | | | |
|------|--------|------|------|--------------------------------|
| 087A | CROUT | | | |
| 087A | OE0D | MVI | C,CR | Load CARRIAGE RETURN code in C |
| 087C | CD3E08 | CALL | ECHO | Output to console |
| 087F | C9 | RET | | |

| | |
|--------------------|-----------------------|
| NAME OF SUBROUTINE | MSGOUT |
| INPUTS | B, H,L |
| OUTPUTS | CHARACTERS TO CONSOLE |
| CALLS | CO |
| DESTROYS | A,C,F/Fs |

DESCRIPTION MSGOUT PRINTS INPUT MESSAGE ON SCREEN. THE ADDRESS OF THE FIRST CHARACTER OF MESSAGE IS CONTAINED IN HL PAIR, AND THE NUMBER OF CHARACTERS IN THE MESSAGE IS CONTAINED IN B.

| | | | | |
|------|--------|------|--------|--|
| 0880 | MSGOUT | | | |
| 0880 | 4E | MOV | C,M | Get character to C |
| 0881 | CD6408 | CALL | CO | Output to console |
| 0884 | 23 | INX | H | Point to next character |
| 0885 | 05 | DCR | B | Decrement number of characters still to be printed |
| 0886 | C28008 | JNZ | MSGOUT | If all characters not printed - repeat |
| 0889 | C9 | RET | | Complete message printed. Therefore return to main programme. |

NAME OF SUBROUTINE TRY
 INPUTS FOUR DIGITS FROM CONSOLE
 OUTPUTS MESSAGE ON SCREEN
 CALLS MSGOUT, SRCH, GETCMD
 DESTROYS A,B,C,E,H,L,F/Ps

DESCRIPTION TRY CHECKS UP IF THE DIAGONSTIC PROGRAMME, FOR THE IC USER WANTS TO TEST, IS AVAILABLE OR NOT. IN CASE THE PROGRAMME IS AVAILABLE, THE AVAILABILITY IS CONFIRMED AND THE STARTING ADDRESS FOR THAT PROGRAMME IS PRINTED ON THE SCREEN OTHERWISE 'NOT AVAILABLE' MESSAGE IS PRINTED.

| | | | | |
|------|--------|------|----------|--|
| 088A | TRY | | | |
| 088A | 211F0F | LxI | H,TRYMSG | Load memory pointer with first address of TRYMSG |
| 088D | 0609 | MVI | B,LTRMSG | Load the number of characters in TRYMSG IN B |
| 088F | CD8008 | CALL | MSGOUT | Send message to screen |
| 0892 | CD9B08 | CALL | SRCH | Get inputs from console and check availability of programme. |
| 0895 | CD8008 | CALL | MSGOUT | Send result of SRCH to console |
| 0898 | C31608 | JMP | GETCMD | Try for next command |

NAME OF SUBROUTINE SRCH
 INPUTS FOUR DIGITS FROM CONSOLE
 OUTPUTS B,H,L
 CALLS CI, ECHO, ERROR, INPUT, VALDL
 DESTROYS A,B,C,H,L,F/Fs

DESCRIPTION SRCH RECEIVES FOUR DIGITS FROM CONSOLE, CHECKS IF THE PROGRAMME IS AVAILABLE FOR THE IC THESE NUMBERS REPRESENT. THE FIRST ADDRESS OF THE APPROPRIATE MESSAGE IS LOADED IN HL PAIR. THE NUMBER OF CHARACTERS IN THE MESSAGE IS LOADED IN B AND THE PROGRAMME RETURNS BACK TO MAIN PROGRAMME. ANY NUMBER OTHER THAN 74 CAUSES PROGRAMME TO JUMP TO ERROR ROUTINE.

| | | /IN FIRST TWO DIGITS | | |
|------|--------|----------------------|-------|---|
| 089B | SRCH | | | |
| 089B | CD5708 | CALL | CI | Get character from console |
| 089E | CD3E08 | CALL | ECHO | Echo it on the screen |
| 08A1 | 79 | MOV | A,C | Move character to A |
| 08A2 | FE37 | CPI | '7' | Is it 7? |
| 08A4 | C26F08 | JNZ | ERROR | No - Illegal command |
| 08A7 | CD5708 | CALL | CI | Yes - Get next character |
| 08AA | CD3E08 | CALL | ECHO | Echo it on the screen |
| 08AD | 79 | MOV | A,C |] Is it 4? |
| 08AE | FE34 | CPI | '4' | |
| 08B0 | C26F08 | JNZ | ERROR | No - Illegal command |
| 08B3 | CDFF08 | CALL | INPUT | Get next character, a valid digit |
| 08B6 | 07 | RLC | A |] Move digit to most significant four bits in A |
| 08B7 | 07 | RLC | A | |
| 08B8 | 07 | RLC | A | |
| 08B9 | 07 | RLC | A | |
| 08BA | 5A | MOV | E,A | Store digit in E |
| 08BB | CDFF08 | CALL | INPUT | Get next digit |
| 08BE | 83 | ADD | E | Place digit in last four bits of A. |

| | | | | |
|------|--------|------|----------|---|
| 08BF | 5F | MOV | E,A | Restore result in E |
| 08C0 | CD5708 | CALL | CI | Get delimiter |
| 08C3 | CD3E08 | CALL | ECHO | Echo it on the screen |
| 08C6 | CD3E09 | CALL | VALDL | Check for correct delimiter |
| 08C9 | D26F08 | JNC | ERROR | Wrong delimiter - Illegal command |
| 08CC | 7B | MOV | A,E | Get digits inputted to A |
| 08CD | FE00 | CPI | 7400 | Was the IC specified 7400? |
| 08CF | C2D808 | JNZ | SRCH05 | No-Branch |
| 08D2 | 21280F | LxI | H,7400MG | Yes-Load memory pointer with the first address of 7400 MG. |
| 08D5 | 0617 | MVI | B,L7400M | Load B with number of character in 7400MG |
| 08D7 | C9 | RET | | |
| 08D8 | SRCH05 | | | |
| 08D8 | FE76 | CPI | 7476 | Was the IC specified 7476? |
| 08DA | C2E308 | JNZ | SRCH10 | No-Branch |
| 08DD | 213F0F | LxI | H,7476MG | Yes - Load memory pointer with the first address of 7476MG. |
| 08E0 | 0617 | MVI | B,L7476M | Load B with number of characters in 7476MG |
| 08E2 | C9 | RET | | |
| 08E3 | SRCH10 | | | |
| 08E3 | FE90 | CPI | 7490 | Was the IC specified 7490? |
| 08E5 | C2EE08 | JNZ | SRCH15 | No-Branch |
| 08E8 | 21560F | LxI | H,7490MG | Yes-Load memory pointer with the first address of 7490MG |
| 08EB | 0617 | MVI | B,L7490M | Load B with number of characters in 7490MG |
| 08ED | C9 | RET | | |
| 08EE | SRCH15 | | | |

| | | | | |
|------|--------|--------|----------|---|
| 08EE | FE93 | CPI | 7493 | Was the IC specified 7493? |
| 08F0 | C2F908 | JNZ | SRCH20 | No-Branch |
| 08F3 | 216D0F | LxI | H,7493MG | Yes-Load memory pointer with the first address of 7493MG |
| 08F6 | 0617 | MVI | B,L7493M | Load B with the number of characters in 7493MG |
| 08F8 | C9 | RET | | |
| 08F9 | | SRCH20 | | |
| 08F9 | 21840F | LxI | H,NTAVMG | Search failed. Load memory pointer with the first address of NTAVMG |
| 08FC | 0611 | MVI | B,LNTAVM | Load B with number of characters in NTAVMG |
| 08FE | C9 | RET | | |

NAME OF SUBROUTINE INPUT
 INPUTS CHARACTER FROM CONSOLE
 OUTPUTS C- CHARACTER FROM CONSOLE
 CALLS CI, ECHO, VALDG, ERROR, CNVBN
 DESTROYS A,B,C,F/Fs

DESCRIPTION INPUT GETS A CHARACTER FROM CONSOLE.
 IF IT IS A VALID DIGIT, THE PROGRAMME
 CONVERTS IT TO BINARY AND RETURNS.
 OTHERWISE CONTROL IS TRANSFERED TO
 ERROR.

| | | | | |
|-----|--------|-------|-------|------------------------------|
| 8FF | | INPUT | | |
| 8FF | CD5708 | CALL | CI | Input character from console |
| 902 | CD3E08 | CALL | ECHO | Echo it on screen |
| 905 | CD1009 | CALL | VALDG | See if valid digit |
| 908 | D25F08 | JNC | ERROR | No - Illegal command |
| 90B | CD3509 | CALL | CNVBN | Yes - convert to binary |
| 90E | C9 | RET | | |

NAME OF SUBROUTINE VALDG
 INPUTS C
 OUTPUTS NONE
 CALLS NOTHING
 DESTROYS A,F/Fs

DESCRIPTION VALDG CHECKS IF THE CHARACTER IN C IS A VALID HEXA-DIGIT. FOR A VALID HEXA-DIGIT THE CONTROL IS TRANSFERED TO SRET AND FOR INVALID HEXA-DIGIT TO FRET.

| | | | | |
|------|--------|-----|------|---|
| 0910 | VALDG | | | |
| 0910 | 79 | MOV | A,C |] Test character against '0' |
| 0911 | FE30 | CPI | '0' | |
| 0913 | FA3209 | JM | FRET | If ASCII code is less, it cannot be a valid digit. Branch |
| 0916 | FE39 | CPI | '9' | Else see if in range 0-9 |
| 0918 | FA3009 | JM | SRET | Code between 0 and 9 |
| 0918 | CA3009 | JZ | SRET | Code equals 9 |
| 091E | FE41 | CPI | 'A' | Not a digit - Try for a better |
| 0920 | FA3209 | JM | FRET | No - code between 9 and A |
| 0923 | FE47 | CPI | 'G' |] No - code greater than F |
| 0925 | F23209 | JP | FRET | |
| 0928 | C33009 | JMP | SRET | O.K. - code is A to F, inclusive |

NAME OF SUBROUTINE SRET
 INPUTS NONE
 OUTPUT CARRY FLAG
 CALLS NOTHING
 DESTROYS CARRY

DESCRIPTION SRET SETS CARRY AND RETURNS

| | | | |
|------|------|-----|----------------|
| 0930 | SRET | | |
| 0930 | 37 | STC | Set carry TRUE |
| 0931 | C9 | RET | |

NAME OF SUBROUTINE FRET
 INPUTS NONE
 OUTPUTS CARRY FLAG
 CALLS NOTHING
 DESTROYS CARRY
 DESCRIPTION FRET RESETS CARRY AND RETURNS

0932 FRET
 0932 37 STC Set carry TRUE
 0933 3F CMC Set if FALSE
 0934 C9 RET

NAME OF SUBROUTINE CNVBN
 INPUTS C- Digit in ASCII CODE
 OUTPUTS A - NUMBER IN BINARY
 CALLS NOTHING
 DESTROYS A, F/Fs
 DESCRIPTION CNVBN CONVERTS THE ASCII CODE IN/TO^c
 BINARY. IT DOES NOT CHECK FOR CORRECTNESS
 OF ASCII INPUT.

0935 CNVBN
 0935 79 MOV A,C
 0936 D630 SUI ZERO] Subtract code for '0' from
 argument
 0938 FEOA CPI '10' Want to test for result of
 0 to 9.
 093A F8 RM If so, Return
 093B D607 SUI '7' Else, return after subtrac-
 ting a bias of 7.
 093D C9 RET

NAME OF SUBROUTINE VALDL
 INPUTS C- CHARACTER FROM CONSOLE
 OUTPUTS NONE
 CALLS NOTHING
 DESTROYS A,F/Fs

DESCRIPTION VALDL CHECKS DELIMITER. IF IT IS
 CARRIAGE RETURN, VALDL TRANSFERS
 CONTROL TO SRET, OTHERWISE CONTROL
 IS TRANSFERED TO FRET

093E VALDL

| | | | | |
|------|--------|-----|------|-----------------------------|
| 093E | 79 | MOV | A,C |] Check for CARRIAGE RETURN |
| 093F | FEOD | CPI | CR | |
| 0941 | CA3009 | JZ | SRET | Found |
| 0944 | C33209 | JMP | FRET | Not found |

NAME OF SUBROUTINE LIST
 INPUTS NONE
 OUTPUTS MESSAGE ON SCREEN
 CALLS CI, ECHO, VALDL, MSGOUT,
 DESTROYS A,B,C,F/Fs

DESCRIPTION IN RESPONSE TO A VALID LIST COMMAND
 LIST DISPLAYS THE LIST OF ALL ICS
 FOR WHICH PROGRAMMES ARE AVAILABLE
 AND TRANSFERS CONTROL TO GET CMD

0947 LIST

| | | | | |
|------|-----------------------|------|---------|---|
| 0947 | CD5708 | CALL | CI | Get input from console |
| 094A | CD3E08 | CALL | ECHO | Echo to terminal |
| 094D | CD3E09 | CALL | VALDL | Check for correct De-limiter |
| 0950 | D26F08 | JNC | ERROR | Not found - Illegal command |
| 0953 | ²¹ 950F | LxI | H,LSTMG | Found -Load memory pointer with the first address of LSTMG |
| 0956 | 0637 | MVI | B,LSTM | Load number of characters in LSTMG in B |
| 0958 | CD8008 | CALL | MSGOUT | Display message on screen |
| 095B | C31608 | JMP | GETCMD | Try for another command |

NAME OF SUBROUTINE GCMD
 INPUTS FOUR HEXA-DIGITS FROM CONSOLE
 OUTPUTS NONE
 CALLS GETHX, ERROR
 DESTROYS A,B,C,D,E,H,L,F/Fs

DESCRIPTION GCMD RECEIVES THE STARTING ADDRESS OF PROGRAMME TO BE EXECUTED FROM CONSOLE. IT LOADS THE ADDRESS ON PROGRAMME COUNTER AND THE CONTROL IS TRANSFERED TO THE PROGRAMME TO BE EXECUTED. IF THE COMMAND IS TERMINATED BY CARRIAGE RETURN, GCMD WILL LOAD WHATEVER IS THE CONTENT OF MEMORY LOCATION OFDEH AND OFDFH IN PROGRAMME COUNTER

| | | | | |
|------|--------|------|---------|--|
| 095E | GCMD | | | /WITHOUT SPECIFYING ANY DIGITS |
| 095E | CD7D09 | CALL | GETHX | Get starting address from console |
| 0961 | D27B09 | JNC | GCM05 | No address specified-Branch |
| 0964 | 7A | MOV | A,D | Get De-limiter |
| 0965 | FE0D | CPI | CR |] If de-limiter not correct- Illegal command |
| 0967 | C26F08 | JNZ | ERROR | |
| 096A | 21DE0F | LxI | H,PADDR | Load programme's starting address in HL |
| 096D | 71 | MOV | M,C | Lower address byte in memory |
| 096E | 23 | INX | H | Point to next location |
| 096F | 70 | MOV | M,B | Store higher address byte in memory |
| 0970 | C37909 | JMP | GCM10 | Branch |
| 0973 | GCM05 | | | |
| 0973 | 7A | MOV | A,D |] No address specified. Check for correct delimiter. |
| 0974 | FE0D | CPI | CR | |
| 0976 | C26F08 | JNZ | ERROR | Wrong delimiter - Illegal command |
| 0979 | GCM10 | | | |
| 0979 | 2ADE0F | LHLD | OFDEH | Get the starting address in HL |
| 097C | E9 | PCHL | | Load it in programme counter |

NAME OF SUBROUTINE GETHX
 INPUTS CHARACTERS FROM CONSOLE
 OUTPUTS BC - 16BIT INTEGER, D-DELIMITER
 CARRY - 1 IF FIRST CHARACTER WAS NOT DELIMITER
 - 0 IF FIRST CHARACTER WAS DELIMITER

CALLS CI, ECHO, VALDL, VALDG, CNVBN
 ERROR

DESTROYS A, B, C, D, E, F/Fs

DESCRIPTION GETHX ACCEPTS A STRING OF CHARACTERS FROM INPUT STREAM AND RETURNS THEIR VALUE AS A 16BIT BINARY INTEGER. IF MORE THAN 4 HEX DIGITS ARE ENTERED, ONLY THE LAST FOUR ARE USED. THE NUMBER TERMINATES WHEN A VALID DELIMITER IS ENCOUNTERED. THE DELIMITER IS ALSO RETURNED AS AN OUTPUT. ILLEGAL CHARACTERS (NOT HEX DIGITS OR DELIMITERS) CAUSE AN ERROR INDICATION. IF THE FIRST (VALID) CHARACTER ENCOUNTERED IN THE INPUT STREAM IS NOT A DELIMITER, CARRY BIT WILL BE SET TO 1. OTHERWISE, CARRY BIT IS SET TO ZERO AND THE CONTENTS OF BC ARE UNDEFINED.

| | | | | |
|------|--------|--------|--------|--|
| 097D | | GETHX | | |
| 097D | E5 | PUSH | H | Save HL |
| 097E | 210000 | LXI | H,0 | Initialize result |
| 0981 | 1E00 | MVI | E,0 | Initialize digit flag to false |
| 0983 | | GTHX05 | | |
| 0983 | CD5708 | CALL | CI | Get a character |
| 0986 | CD3E08 | CALL | ECHO | Echo it on screen |
| 0989 | CD3E09 | CALL | VALDL | See if delimiter |
| 098C | D29B09 | JNC | GTHX10 | No-branch |
| 098F | 51 | MOV | D,C | Yes - All done, but want to return delimiter |
| 0990 | E5 | PUSH | H |] Move result to BC |
| 0991 | C1 | POP | B | |

| | | | | |
|------|--------|------|--------|---|
| 0992 | E1 | POP | H | Restore HL |
| 0993 | 7B | MOV | A,E | Get flag |
| 0994 | B7 | ORA | A | Set F/Fs |
| 0995 | C23009 | JNZ | SRET | If flag non zero, a number has been found |
| 0998 | CA3209 | JZ | FRET | Else, Delimiter was first Character |
| 099B | GTHX10 | | | |
| 099B | CD1009 | CALL | VALDG | If not delimiter, see if digit |
| 099E | D26F08 | JNC | ERROR | Error if not a valid digit, either |
| 09A1 | CD3509 | CALL | CNVBN | Convert digit to its binary value |
| 09A4 | 1EFF | MVI | E,FFH | SET digit flag non-zero |
| 09A6 | 29 | DAD | H | X2 |
| 09A7 | 29 | DAD | H | X4 |
| 09A8 | 29 | DAD | H | X8 |
| 09A9 | 29 | DAD | H | X16 |
| 09AA | 0600 | MVI | B,0 | Clear upper 8 bits of BC pair |
| 09AC | 4F | MOV | C,A | Binary value of character in C |
| 09AD | 09 | DAD | B | Add this value to partial result |
| 09AE | C38309 | JMP | GTHX05 | Get next character |

MONITOR TABLES

| | | | | | |
|------|----------|--------|-----|------|---|
| OF00 | | MSG | | | Sign on Message |
| OF00 | 0DOA492E | | DB | | |
| OF04 | 432E5445 | | | | |
| OF08 | 53544552 | | | | |
| OF0C | 20524541 | | | | |
| OF10 | 44590DOA | | | | |
| 0014 | | LMSG | EQU | 14H | Length of MSG |
| OF14 | | CTAB | | | Table of address of command routines |
| OF14 | 54 | | DB | 'T' | |
| OF15 | 47 | | DB | 'G' | |
| OF16 | 4C | | DB | 'L' | |
| 0003 | | NCMDS | EQU | 03H | Number of valid commands |
| OF17 | | CADDR | | | Table of addresses of functional commands |
| OF17 | 0000 | | DW | 0 | Dummy |
| OF19 | 4709 | | DW | LIST | |
| OF1B | 5E09 | | DW | GCMD | |
| OF1D | 8A08 | | DW | TRY | |
| OF1F | | TRYMSG | | | Message in response to T command |
| OF1F | ODOA492E | | DB | | LF,CR,I,., |
| OF23 | 43234E4F | | | | C,.,N,O, |
| OF27 | 3A | | | | ., |
| 0009 | | LTRMSG | EQU | 09H | Length of TRYMSG |

| | | | | |
|------|----------|-----|-----|---|
| OF28 | 7400MG | | | Message for 7400 IC testing programme location |
| OF28 | ODOA4156 | DB | | LF,CR,A,V, |
| OF2C | 41494C41 | | | A,I,L,A, |
| OF30 | 424C450D | | | B,L,E,LF, |
| OF34 | 0A4C4F43 | | | CR,L,O,C, |
| OF38 | 3A303942 | | | : ,O,9,B |
| OF3C | 310DOA | | | 1,LF,CR |
| 0017 | L7400M | EQU | 17H | Length of 7400 MG |
| OF3F | 7476MG | | | Message for 7476IC testing programme location |
| OF3F | ODOA4156 | DB | | LF,CR,A,V |
| OF43 | 41494C41 | | | A,I,L,A, |
| OF47 | 424C450D | | | B,L,E,LF, |
| OF4B | 0A4C4F43 | | | CR,L,O,C, |
| OF4F | 3A304137 | | | : ,O,A,7, |
| OF53 | 350DOA | | | 5,LF,CR |
| 0017 | L7476M | EQU | 17H | Length of 7476MG |
| OF56 | 7490MG | | | Message for 7490IC testing programme location |
| OF56 | ODOA4156 | DB | | LF,CR,A,V, |
| OF5A | 41494C41 | | | A,I,L,A, |
| OF5E | 424C450D | | | B,L,E,LF, |
| OF62 | CA4C4F43 | | | CR,L,O,C, |
| OF66 | 3A304330 | | | : ,O,C,O |
| OF6A | 300 DOA | | | O,LF,CR |

| | | | | |
|------|----------|-----|-----|---|
| 0017 | L7490M | EQU | 17H | Length of 7490MG |
| OF6D | 7493MG | | | Message for 7493IC testing programme location |
| OF6D | OD0A4156 | DB | | LF,CR,A,V, |
| OF71 | 41494C41 | | | A,I,L,A, |
| OF75 | 424C450D | | | B,L,E,LF |
| OF79 | 0A4C4F43 | | | CR,L,O,C, |
| OF7D | 3A304335 | | | .,O,C,5, |
| OF81 | 450DOA | | | E,LF,CR |
| 0017 | L7493M | EQU | 17H | Length of 7493MG |
| OF84 | NTAVMG | | | Message for test programme not available. |
| OF84 | OD0A4E4F | DB | | LF,CR,N,O, |
| OF88 | 54204156 | | | T,SP,A,V, |
| OF8C | 41494C41 | | | A,I,L,A, |
| OF90 | 424C450D | | | B,L,E,LF, |
| OF94 | OA | | | CR |
| 0011 | LNTAVM | EQU | 11H | Length of NTAVMG |
| OF95 | LSTMG | | | Message for L command |
| OF95 | OD0A464F | DB | | LF,CR,F,O |
| OF99 | 4C4C4F57 | | | L,L,O,W, |
| OF9D | 494E4720 | | | T,N,G,SP, |
| OFA1 | 43414E20 | | | C,A,N,SP, |

| | | | | |
|------|----------|-----|-----|-----------------|
| OFA5 | 42452054 | | | B,E,SP,T, |
| OFA9 | 45535445 | | | E,S,T,E, |
| OFAD | 44ODOA20 | | | D,LF,CR,SP, |
| OFB1 | 37343030 | | | 7,4,0,0, |
| OFB5 | ODOA2037 | | | LF,CR,SP,7, |
| OFB9 | 3437360D | | | 4,7,6,LF, |
| OFBD | 0A203734 | | | CR,SP,7,4, |
| OFC1 | 3930ODOA | | | 9,0,LF,CR, |
| OFC5 | 20373439 | | | SP,7,4,9, |
| OFC9 | 33ODOA | | | 3,LF,CR |
| 0037 | LLSTM | EQU | 37H | Length of LSTMG |

MONITOR EQUATES

| | | | | |
|------|-------|-----|-------|--|
| 0027 | CMD | EQU | 27H | Command instruction for USART initialization. |
| 00E1 | CNTL | EQU | E1H | USART control port. |
| 000D | CR | EQU | 0DH | ASCII code for CARRIAGE RETURN |
| 00E0 | DATA | EQU | E0H | USART data port. |
| 001B | ESC | EQU | 1BH | ASCII code for ESCAPE character |
| 0FFF | ICNO | EQU | 0FFFH | Location for 'IC test programme' indicator |
| 000A | LF | EQU | 0AH | ASCII code for LINE FEED |
| 00CE | MODE | EQU | CEH | Mode set for USART initialization |
| 0FDE | PADDR | EQU | 0FDEH | Location for test programme starting address in GCMD sub- routine. |
| 007F | PRTYO | EQU | 7FH | Mask to clear parity bit from console character. |
| 0FFE | RPTLP | EQU | 0FFEH | Location for 'Repeat Loop Indicator' address. |
| 0030 | ZERO | EQU | 30H | ASCII code for '0'. |

4.10 OPERATING PROCEDURE FOR USING ADM-3A KEYBOARD

The procedure recommended for communicating with the HIL-2961 through ADM-3A keyboard is as under:

1. Before connecting the hardware interface between ADM-3A and HIL-2961, switch all power supplies OFF, including HIL-2961 and ADM-3A supplies.
2. Connect the hardware.
3. Switch ON all power supplies for hardware interface.
4. Switch ON ADM-3A and HIL-2961 supplies.
5. Load monitor programme in microprocessor RAM through HIL-2961 keyboard, beginning from location 0800H.
6. Execute programme from location 0800H through HIL-2961 keyboard 'IC. TESTER IS READY' will be displayed on ADM-3A screen.
7. Control has been transferred to I.C. tester monitor ADM-3A keyboard is now operative. Enter all valid commands through ADM-3A keyboard, developed for IC Tester.
8. If, at any stage the user wants to revert to HIL-2961 keyboard, RESET key on HIL-2961 keyboard should be pressed.
9. Having reverted to HIL-2961, the user may once again transfer control to ADM-3A keyboard by executing programme from location 0808H (NOT FROM 0800H) through HIL-2961 keyboard. Memory location 0800H to 0807H contain the initialization programme for 8251A USART and must not be repeated once the USART is initialized.

CHAPTER - 5

TEST PROGRAMMES DEVELOPMENT FOR A FEW SELECTED IC CHIPS

5.1 INTRODUCTION

Test programme for IC 7400 was discussed in Chapter 2. This programme was executed through HIL-2961 keyboard. The limitations of operating through HIL-2961 keyboard have been covered under Section 3.1. Having developed a successful interface between the ADM-3A and HIL-2961, test programmes for different ICs can now be executed through ADM-3A keyboard.

The versatility of an IC tester depends upon its capability to test a wide range of ICs. However, if the IC tester is developed for use in a particular laboratory using certain specific ICs, it is sufficient that the IC tester be able to test those ICs. In this dissertation, four IC have been selected for developing test programmes. These are 7400, 7476, 7490 and 7493. Of these, the first two are SSI chips and the remaining two are MSI chips.

5.2 TEST PROGRAMMES

Test programme for IC 7400 has been modified to enable its execution through the ADM-3A keyboard. Development of the test programmes for different ICs, listed in 5.1 is

discussed individually later in this chapter. All the test programmes are given together at the end of the chapter.

These test programmes make use of the existing ports on the HIL 2961, viz., Port 30 (I/P port) and Port 20 (I/P port). Data lines of these ports are connected to two PCB edge connectors. To facilitate identification, I/P data bus uses RED flexible wires and O/P data bus uses BLUE flexible wires. These wires are appropriately labelled 0,1,2,... etc. to indicate that they represent $D_0, D_1, D_2 \dots$ etc. data lines. The PCB edge connectors also carry a terminal each, for V_{cc} and Ground.

5.3 TESTING PROCEDURE

The memory location of the starting address for a particular IC testing programme can be found by using T command (TRY Subroutine). The execution of the test programme, after connecting the appropriate IC test card, is accomplished by using G command (GCMD Subroutine). If the IC is good, 'I.C. IS GOOD' will be displayed on the screen and the green LED will glow on the test card. If the IC under test has any logic mal-functions, 'I.C. IS BAD' will be displayed on the ADM-3A screen, a horn will be sounded and the red LED on IC test card will glow. The control is not yet transferred to the IC tester monitor. In case the user

wants to test the IC repeatedly for reliability testing, ~~if any~~, R key should be pressed. At present, this will cause the programme to be repeated ten times. However, the programme may be repeated desired number of times with a minor change in the test programme. Once the user is satisfied with the testing and wants to end the testing procedure, S key should be pressed. This causes the control to be transferred to the IC tester monitor, in preparation for the next test.

The recommended procedure for testing is as follows:

1. Activate the ADM-3A to HIL-2961 interface.
2. Load all the IC test programmes given at the end of this chapter into their respective memory locations through HIL-2961 keyboard. Now, transfer the control to ADM-3A keyboard.
3. Use TRY command to find the starting address of the IC test programme required.
4. Connect the IC tester card between the two PCB edge connectors specified in Section 5.2.
5. Connect the V_{cc} and GND terminals of PCB edge connectors appropriately to +5V and GND on power supply unit.
6. Using G command execute the test programme.
7. If the IC is declared good, the test may be repeated 10 times for reliability testing by pressing 'R' key on ADM-3A key board.

8. To exit from the programme, press 'S' key. 'I.C.TESTER
 ■ READY' will be displayed on the screen and the control
 is transferred to the IC tester monitor.
9. The IC tester is once again ready to test another IC.

Note: The user must connect the IC test card between the two PCB edge connectors before executing the test programme.

5.4 IC 7400 TEST PROGRAMME

Two basic changes have been made in the IC 7400/^{test}programme discussed in Chapter 2. The first one is the alteration to use the ADM-3A capability of printing messages on the screen and the second one is creation of a number of sub-routines which can be used by other programmes as well. The flow chart is given in Fig.5.1. The programme for IC 7400 test is labelled NDTST and starts at memory location 09B1H. A generalised flow chart describing the test procedure, in general, for all ICs is given in Fig.5.2.

5.5 IC 7476 TEST PROGRAMME

7476 is a dual J-K F/F with independent Preset, clear and clock. Preset and clear are active LOW asynchronous inputs and have precedence over the synchronous inputs i.e., J, K and Clock. Once both the asynchronous inputs are disabled, the J&K terminals become active. In response to a particular set of J-K inputs, the correct output shall appear on the output terminals at the falling edge of the clock. This is a

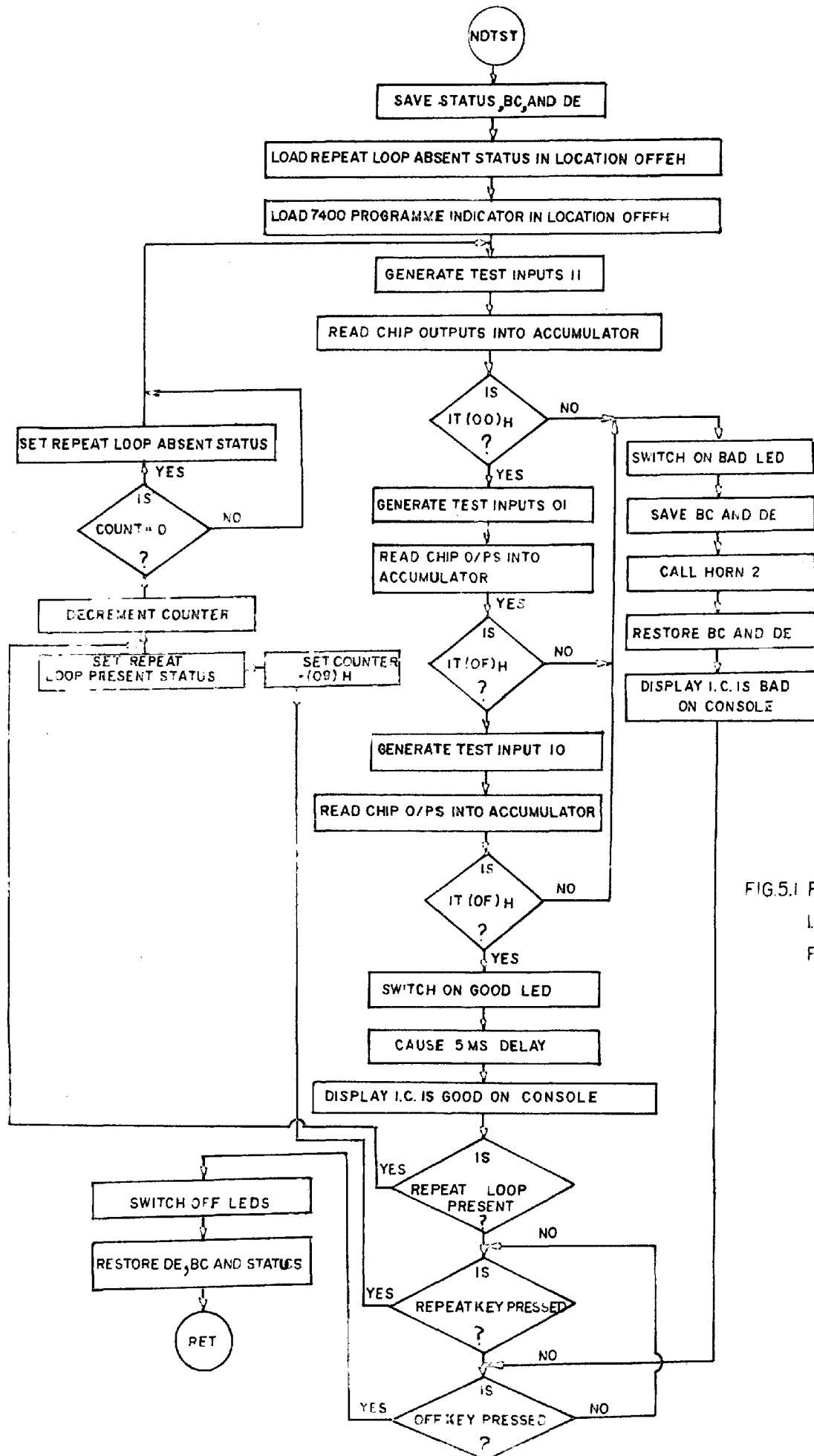
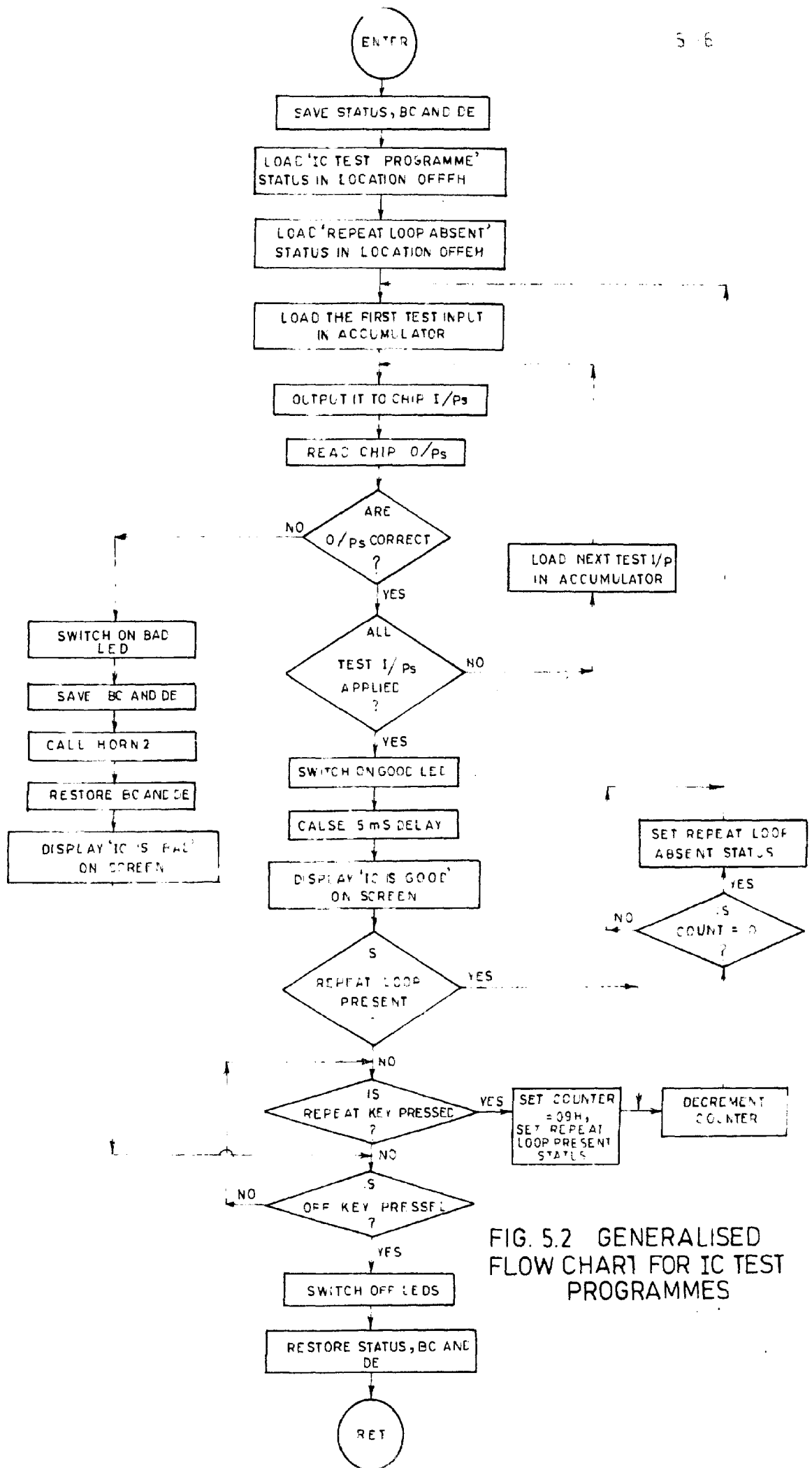


FIG.5.1 FLOW CHART FOR IC.7400 TEST PROGRAMME.



5-5

FIG. 5.2 GENERALISED FLOW CHART FOR IC TEST PROGRAMMES

very important point to be taken care of, in testing the IC chip. In the test programme developed, in order to cater for this feature, each test input vector is first applied with clock (Bit D_2 of O/P port) input HIGH. The chip outputs at this stage will retain their previous state, for a good IC. Keeping the test inputs unchanged, the clock input terminal is made LOW and the chip outputs are compared to the correct output desired in response to the test input vector currently in use.

7476 is a SSI device. Its pin configuration and functional table is given in Appendix 'E'. The circuit diagram for 7476 card is shown in Fig.5.3 and the PCB layout is shown in Fig.5.4. The programme^{for}/IC 7476 test is labelled 'JKFF' and starts from memory location OA75H.

5.6 IC 7490 TEST PROGRAMME

7490 is a divide by two and divide by five, 4 bit binary up counter IC. It can be used as a divide by ten counter by connecting output of the first F/F (Q_A) to clock terminal of the second F/F (I/P B). It has four asynchronous inputs which have precedence over all other inputs. These are $R_0(1)$, $R_0(2)$, $R_g(1)$ and $R_g(2)$. $R_0(1)$ and $R_0(2)$ are connected inside the chip to the inputs of a two input NAND gate. When both of them are made HIGH, they RESET all the F/Fs. With any other combination, these inputs are disabled. $R_g(1)$ and $R_g(2)$ are also connected internally to the inputs of a two

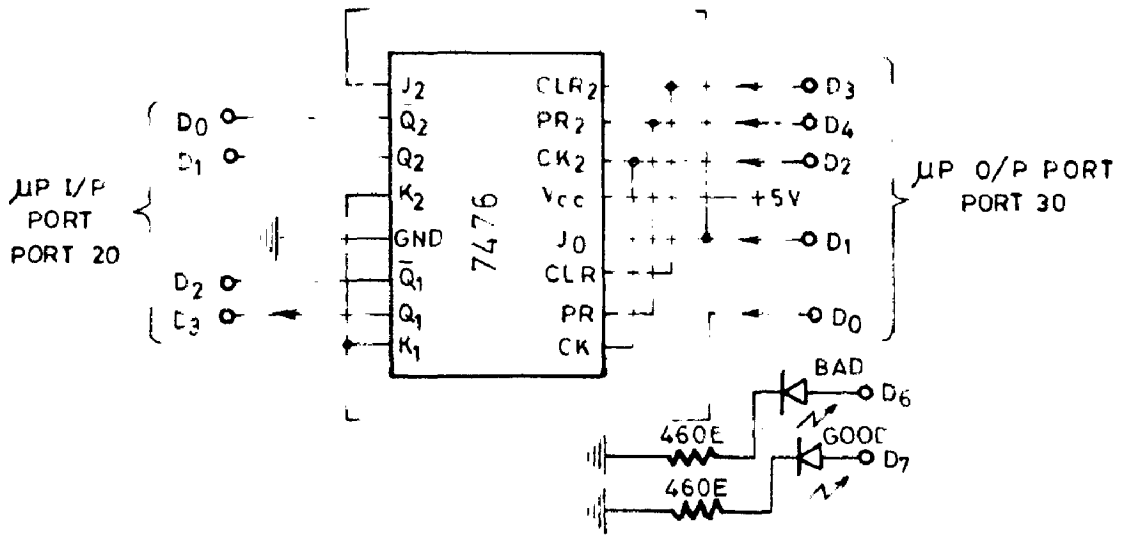


FIG . 5.3 7476 TEST CARD CIRCUIT DIAGRAM

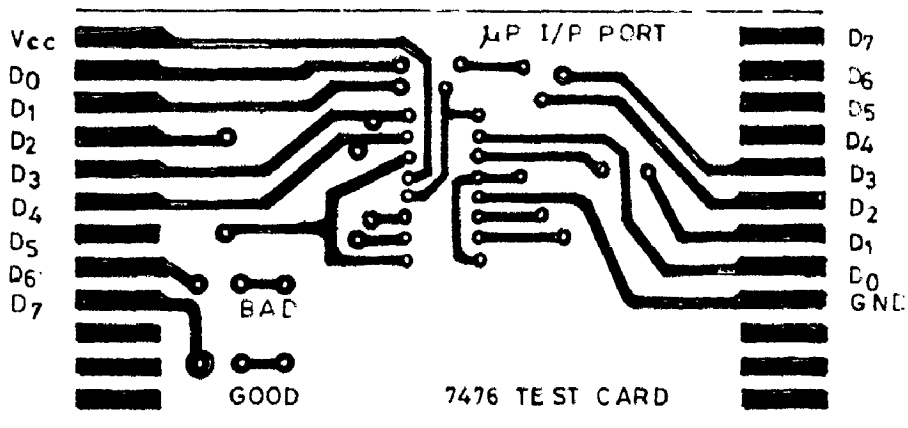


FIG . 5.4 PCB LAYOUT

input NAND gate. When both of them are made HIGH, A and D F/Fs are SET and B and C F/Fs are RESET. Any other combination applied at $R_g(1)$ and $R_g(2)$ terminals disables them. With QA output connected to I/PB input, the IC behaves as a decade up counter. This mode of operation has been used to verify its correct behaviour. Initially, the effect of asynchronous inputs is checked. Then the count is started from zero count and is made to complete one complete count sequence returning back to the zero count. Like 7476, ICs 7490 and 7493 also place the desired output in response to a particular input at the falling edge of the clock pulse. Therefore, the testing procedure is accordingly designed as described in Section 5.5.

7490 is a MSI device. Its details are given in Appendix 'E'. The circuit diagram for 7490 test card is given in Fig.5.5 and the PCB layout is given in Fig.5.6. The programme for IC 7490 test is labelled T90 and starts from memory location 0C00H.

5.7 IC 7493 TEST PROGRAMME

7493 is basically a divide by two and divide by eight 4 bit binary up ripple counter. It can be used as a divide by 16 counter by connecting the output of the first F/F (Q_A) to the clock terminal of the second F/F (I/P B). It has two asynchronous inputs, $R_0(1)$ and $R_0(2)$ which have precedence over other inputs. $R_0(1)$ and $R_0(2)$ are connected internally

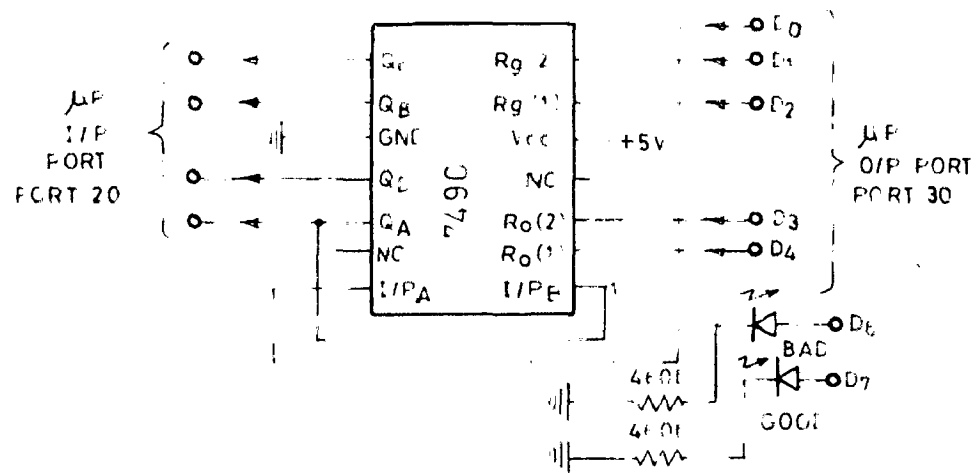


FIG. 5.5 7490 TEST CARD CIRCUIT DIAGRAM

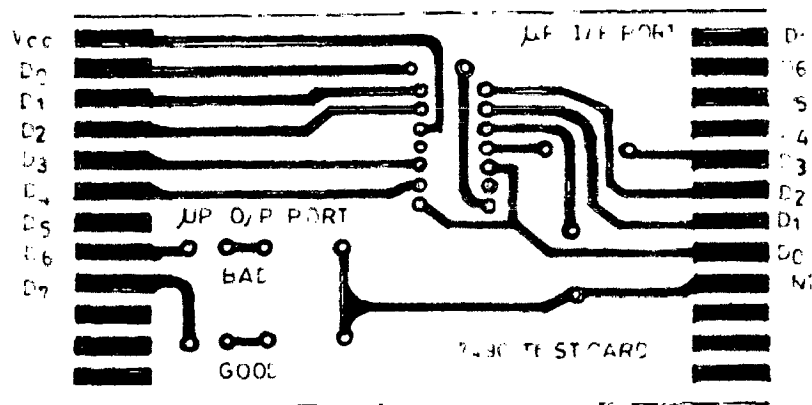


FIG. 5.6 PCB LAYOUT

to the inputs of a two input NAND gate, the output of which goes to the clear terminals (not available externally) of all the four F/Fs. Thus, when both $R_0(1)$ and $R_0(2)$ are held HIGH, all the F/Fs are RESET. Any other combination disables these inputs.

With Q_A output connected to I/P B input, the IC behaves as a divide by sixteen up counter. This mode of operation has been used to verify its correct behaviour. It can be seen that IC 7493 is quite similar to IC 7490, therefore the testing procedure for the two is similar. Only difference in this case is that the count goes upto fifteen after starting from count zero.

7493 is a MSI device. Its details are given in Appendix 'E'. The circuit diagram ^{for} 7493 test card is given in Fig.5.7 and the PCB layout is given in Fig.5.8.

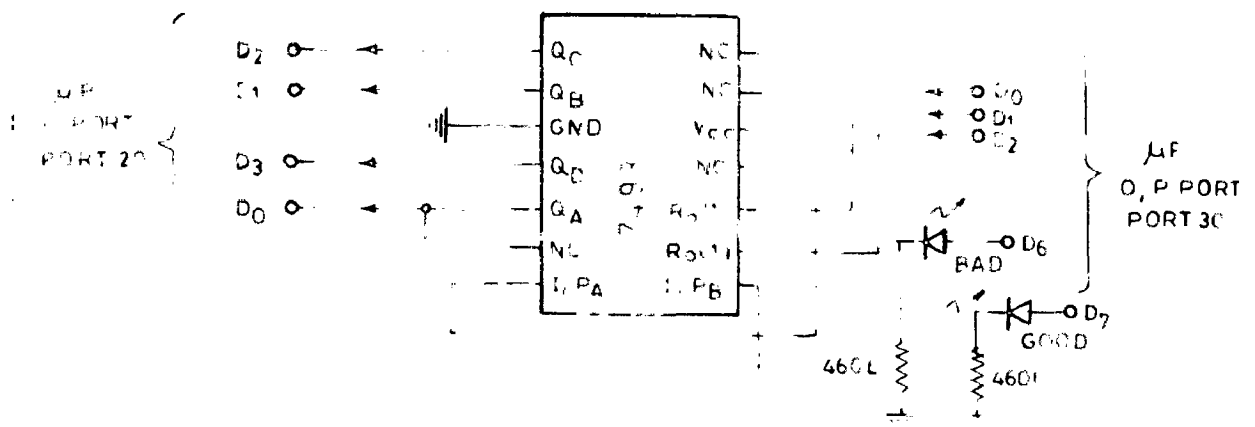


FIG. 5.7 7493 TEST CARD CIRCUIT DIAGRAM

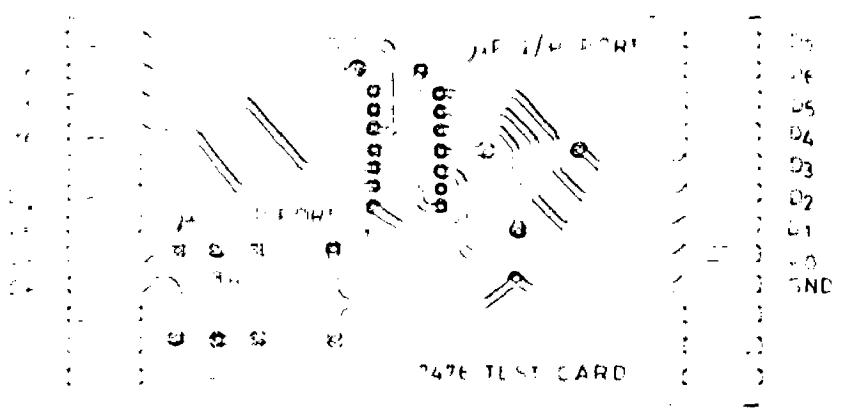


FIG. 5.8 PCB LAYOUT

5.8 PROGRAMME LISTING

| | | NAME OF SUBROUTINE | NDTST | |
|------|--------|--------------------|---------------------|---|
| | | INPUTS | NONE | |
| | | OUTPUTS | NONE | |
| | | CALLS | LOAD | |
| | | DESTROYS | H,L,F/Fs | |
| | | DESCRIPTION | 'NDTST' | CHCKS UPIC 7400 |
| 09B1 | | NDTST | | |
| 09B1 | F5 | PUSH | PSW | Save status |
| 09B2 | C5 | PUSH | B | Save BC pair |
| 09B3 | D5 | PUSH | D | Save DE pair |
| 09B4 | 3E00 | MVI | A,(00) _H | Load REPEAT LOOP ABSENT status |
| 09B6 | 32FEOF | STA | RPTLP | |
| 09B9 | 32FFOF | STA | ICNO | Load 7400 Test Programme indicator |
| 09BC | | ND5 | | |
| 09BC | 3E07 | MVI | A,(07) _H | Send (11) ₂ at NAND gate inputs and read chip O/Ps |
| 09BE | CD0DOA | CALL | LOAD | |
| 09C1 | C2140A | JNZ | FAULT | If chip O/Ps are not correct Branch |
| 09C4 | 3E06 | MVI | A,(06) _H | Send (10) ₂ at NAND gate inputs and read chip O/Ps |
| 09C6 | CD0DOA | CALL | LOAD | |
| 09C9 | FEOF | CPI | (0F) _H | Branch if chip O/Ps are not correct |
| 09CB | C2140A | JNZ | FAULT | |
| 09CE | 3E05 | MVI | A,(05) _H | Send (01) ₂ at NAND gate inputs and read chip O/Ps |
| 09D0 | CD0DOA | CALL | LOAD | |
| 09D3 | FEOF | CPI | (0F) _H | Branch if chip O/Ps are not correct. |
| 09D5 | C2140A | JNZ | FAULT | |
| 09D8 | C3DB09 | JMP | GOOD | The IC is good. Go to Good subroutine. |

NAME OF SUBROUTINE GOOD
 INPUTS NONE
 OUTPUTS MESSAGE ON CONSOLE, SIGNAL FOR SWITCHING
 ON GOOD LED
 CALLS CI, ECHO, MSGOUT, DELB
 DESTROYS A, H, L, F/FB

DESCRIPTION 'GOOD' DISPLAYS 'I.C.IS GOOD'
 ON CONSOLE AND THEN LOOKS FOR
 REPEAT OR STOP COMMANDS

| | | | | | |
|------|--------|------|------|-----------|--------------------------------------|
| 09DB | | GOOD | | | |
| 09DB | 3E80 | | MVI | A, 80H |] Switch ON 'GOOD' LED |
| 09DD | D330 | | OUT | 30H | |
| 09DF | C5 | | PUSH | B | Save BC |
| 09E0 | 010500 | | LXI | B, 0005H |] Cause 5 ms delay |
| 09E3 | CD3004 | | CALL | DELB | |
| 09E6 | 21E00F | | LXI | H, GOODMG |] Display message on console |
| 09E9 | 060F | | MVI | B, LGOODM | |
| 09EB | CD8008 | | CALL | MSGOUT | |
| 09EE | C1 | | POP | B | Restore BC |
| 09EF | 3AFE0F | | LDA | RPTLP |] If repeat loop is present - Branch |
| 09F2 | FE15 | | CPI | 15H | |
| 09F4 | CA3BOA | | JZ | RPT05 | |
| 09F7 | C5 | | PUSH | B | Else, Save BC |
| 09F8 | CD5708 | | CALL | CI | Get a character from console |
| 09FB | CD3E08 | | CALL | ECHO | Echo it |
| 09FE | 79 | | MOV | A, C | Move character to A |
| 09FF | C1 | | POP | B | Restore BC |
| 0A00 | FE52 | | CPI | R | Is character R (REPEAT)? |
| 0A02 | CA330A | | JZ | RPT | Yes - Branch |
| 0A05 | FE53 | | CPI | S | NO - Is it S (STOP)? |
| 0A07 | CA6BOA | | JZ | STOP | Yes - Branch |
| 0A0A | C3F709 | | JMP | GOOD05 | NO - Get another character |

NAME OF SUBROUTINE LOAD
 INPUTS TEST INPUTS IN ACCUMULATOR
 OUTPUTS CHIP OUTPUTS IN ACCUMULATOR
 CALLS NOTHING
 DESTROYS A, F/Fs

DESCRIPTION LOAD SENDS OUT TEST INPUTS TO THE
 IC UNDER TEST AND READS THE IC
 OUTPUTS

| | | | | | |
|------|------|------|-----|-----|--------------------|
| 0A0D | | LOAD | | | |
| 0A0D | D330 | | OUT | 30 | Output test input |
| 0A0F | DB20 | | IN | 20 | Read chip output |
| 0A11 | E60F | | ANI | 0FH | Mask unwanted bits |
| 0A13 | C9 | | RET | | |

NAME OF SUBROUTINE FAULT
 INPUTS NONE
 OUTPUTS MESSAGE ON CONSOLE, HORN AND
 SIGNAL FOR SWITCHING ON 'BAD'
 LED.
 CALLS HORN2, MSGOUT
 DESTROYS A, H, L, F/Fs

DESCRIPTION FAULT SWITCHES ON BAD LED,
 SOUNDS A HORN AND DISPLAYS
 'I.C. IS BAD' ON SCREEN

| | | | | | |
|------|--------|-------|------|----------|------------------------------|
| 0A14 | | FAULT | | | |
| 0A14 | 3E40 | | MVI | A, 40H | Switch 'ON' 'BAD' LED |
| 0A16 | D330 | | OUT | 30 | |
| 0A18 | C5 | | PUSH | B | Save BC |
| 0A19 | D5 | | PUSH | D | Save DE |
| 0A1A | 0620 | | MVI | B, 20H |] Sound Horn |
| 0A1C | 1630 | | MVI | D, 30H | |
| 0A1E | CD4704 | | CALL | HORN2 | |
| 0A21 | 21F00F | | LXI | H, BADMG |] Display message on console |
| 0A24 | 060E | | MVI | B, LBADM | |
| 0A26 | CD8008 | | CALL | MSGOUT | |

| | | | | |
|------|--------|-----|--------|--|
| 0A29 | 3E00 | MVI | A,00H |] Store repeat loop absent status in location OFFEH |
| 0A2B | 32FE0F | STA | RPTLP | |
| 0A2E | D1 | POP | D | Restore DE |
| 0A2F | C1 | POP | B | Restore BC |
| 0A30 | C3F709 | JMP | GOOD05 | Get next character |

| | |
|--------------------|------------|
| NAME OF SUBROUTINE | RPT |
| INPUTS | NONE |
| OUTPUTS | NONE |
| CALLS | NOTHING |
| DESTROYS | A,D,E,F/Fs |

DESCRIPTION RPT SETS UP REPEAT LOOP PRESENT STATUS AND REPEAT LOOP COUNTER. IT THEN DECREASES THE COUNTER. IF THE COUNTER IS EXHAUSTED, IT SETS UP REPEAT LOOP ABSENT STATUS AND RETURNS TO PROGRAMME, OTHERWISE IT RETURNS DIRECTLY TO THE PROGRAMME.

| | | | | |
|------|--------|-------|---------|--|
| 0A33 | | RPT | | |
| 0A33 | 110900 | LXI | D,0009H | Set up repeat loop counter |
| 0A36 | 3E15 | MVI | A,15H |] Set up repeat loop present status |
| 0A38 | 32FE0F | STA | RPTLP | |
| 0A3B | | RPT05 | | |
| 0A3B | 7B | MOV | A,E |] Is lower byte of counter zero? |
| 0A3C | FE00 | CPI | 00H | |
| 0A3E | CA450A | JZ | RPT15 | Yes - Branch |
| 0A41 | | RPT10 | | |
| 0A41 | 1B | DCX | D | No - Decrement counter |
| 0A42 | C3560A | JMP | AGAIN | Repeat test programme |
| 0A45 | | RPT15 | | |
| 0A45 | 7A | MOV | A,D |] Lower byte of counter is zero - Is upper byte also zero? |
| 0A46 | FE00 | CPI | 00H | |
| 0A48 | CA4E0A | JZ | RPT20 | Yes - Branch |

| | | | | | |
|------|--------|-------|-----|--------|--|
| 0A4B | C3410A | | JMP | RPT10 | No - go back to the programme |
| 0A4E | | RPT20 | | | |
| 0A4E | 3E00 | | MVI | A, OOH | Counter is exhausted, store 'repeat loop absent status'. |
| 0A50 | 32FEOF | | STA | RPTLP | |
| 0A53 | C3560A | | JMP | AGAIN | Repeat test programme |

| | |
|--------------------|---------|
| NAME OF SUBROUTINE | AGAIN |
| INPUTS | NONE |
| OUTPUTS | NONE |
| CALLS | NOTHING |
| DESTROYS | A, F/Fs |

DESCRIPTION 'AGAIN' CHECKS UP WHICH IS THE IC TEST PROGRAMME CURRENTLY BEING EXECUTED BY CHECKING THE IC TEST PROGRAMME INDICATOR STORED IN LOCATION OFFFH. CONTROL IS ACCORDINGLY TRANSFERRED TO THAT PROGRAMME.

| | | | | | |
|------|--------|-------|-----|-------|---|
| 0A56 | | AGAIN | | | |
| 0A56 | 3AFF0F | | LDA | ICNO | Load test programme indicator status |
| 0A59 | FE00 | | CPI | OOH | Is it 7400 programme? |
| 0A5B | CABC09 | | JZ | ND5 | Yes - Go to 7400 programme |
| 0A5E | FE76 | | CPI | 76 | No. Is it 7476 programme? |
| 0A60 | CA820A | | JZ | JK05 | Yes - Go to 7476 programme |
| 0A63 | FE90 | | CPI | 90 | No. Is it 7490 programme? |
| 0A65 | CA0DOC | | JZ | T9005 | Yes - Go to 7490 programme |
| 0A68 | C36B0C | | JMP | T9305 | It is 7493 programme |

NAME OF SUBROUTINE STOP
 INPUTS NONE
 OUTPUTS NONE
 CALLS NOTHING
 DESTROYS A,B,C,D,E

DESCRIPTION 'STOP' SWITCH OFF BOTH LEDS AND
 THEN RETURNS TO MONITOR FOR
 GETTING NEXT COMMAND FROM CONSOLE

STOP

| | | | |
|--------|-----|--------|-------------------|
| 3E00 | MVI | A,00H | Switch OFF LEDs |
| D330 | OUT | 30H | |
| D1 | POP | D | Restore DE |
| C1 | POP | B | Restore BC |
| F1 | POP | PSW | Restore status |
| C30808 | JMP | SIGNO1 | Return to Monitor |

NAME OF SUBROUTINE JKFF
 INPUTS NONE
 OUTPUTS NONE
 CALLS LOAD,CHECK1,CHECK2
 DESTROYS H,L,F/Fs

DESCRIPTION 'JKFF' TESTS IC 7476

JKFF

| | | | |
|--------|------|-------|--------------------------------------|
| F5 | PUSH | PSW | Save status |
| C5 | PUSH | B | Save BC |
| D5 | PUSH | D | Save DE |
| 3E76 | MVI | A,76H | Load '7476 Test Programme' status |
| 32FF0F | STA | ICNO | |
| 3E00 | MVI | A,00H | Load 'Repeat Loop Absent' Status |
| 32FE0F | STA | RPTLP | |

| | | | | | |
|------|--------|------|------|-------|---|
| OA82 | | JK05 | | | |
| OA82 | 3E09 | | MVI | A,09H |] Make PRESET LOW and read chip O/Ps |
| OA84 | CDODOA | | CALL | LOAD | |
| OA87 | CDDFOA | | CALL | CHCK1 | Check if O/Ps are correct |
| OA8A | 3E1D | | MVI | A,1DH |] Make J=0, K=1 and read chip O/Ps |
| OA8C | CDODOA | | CALL | LOAD | |
| OA8F | CDDFOA | | CALL | CHCK1 | Check if O/Ps are correct |
| OA92 | 3E19 | | MVI | A,19H |] Make clock ZERO and read chip O/Ps |
| OA94 | CDODOA | | CALL | LOAD | |
| OA97 | CDE50A | | CALL | CHCK2 | Check if O/Ps are correct |
| OA9A | 3E1E | | MVI | A,1EH |] Make J=1, K=0 and read chip O/Ps |
| OA9C | CDODOA | | CALL | LOAD | |
| OA9F | CDE50A | | CALL | CHCK2 | Check if O/Ps are correct |
| OAA2 | 3E1A | | MVI | A,1AH |] Make clock ZERO and read chip O/Ps |
| OAA4 | CDODOA | | CALL | LOAD | |
| OAA7 | CDDFOA | | CALL | CHCK1 | Check if O/Ps are correct |
| OAAA | 3E1F | | MVI | A,1FH |] Make J=1, K=1 and read chip O/Ps |
| OAAC | CDODOA | | CALL | LOAD | |
| OAAF | CDDFOA | | CALL | CHCK1 | Check if O/Ps are correct |
| OAB2 | 3E1B | | MVI | A,1BH |] Make clock ZERO and read chip O/Ps |
| OAB4 | CDODOA | | CALL | LOAD | |
| OAB7 | CDE50A | | CALL | CHCK2 | Check if O/Ps are correct |
| OABA | 3E1C | | MVI | A,1CH |] Make J=0, K=0 and read chip O/Ps |
| OABC | CDODOA | | CALL | LOAD | |
| OABF | CDE50A | | CALL | CHCK2 | Check if O/Ps are correct |
| OAC2 | 3E18 | | MVI | A,18H |] Make clock zero and read chip O/Ps |
| OAC4 | CDODOA | | CALL | LOAD | |
| OAC7 | CDE50A | | CALL | CHCK2 | Check if O/Ps are correct |
| OACA | 3E00 | | MVI | A,00H |] Make PRESET and CLEAR Low and read chip O/Ps |
| OACC | CDODQA | | CALL | LOAD | |

| | | | | |
|------|--------|------|-------|--|
| OACF | FEOF | CPI | OFH | } If O/Ps are not correct Chip is bad |
| OAD1 | C2140A | JNZ | FAULT | |
| OAD4 | 3E12 | MVI | A,12H | } Else, Make CLEAR LOW and read chip O/Ps |
| OAD6 | CDOD0A | CALL | LOAD | |
| OAD9 | CDE50A | CALL | CHCK2 | Check if O/Ps are correct |
| OADC | C3DB09 | JMP | GOOD | Chip is Good |

NAME OF SUBROUTINE CHCK1
 INPUTS CHIP O/Ps IN ACCUMULATOR
 OUTPUTS NONE
 CALLS NOTHING
 DESTROYS F/Ps

DESCRIPTION CHCK1 CHECKS UP IF CHIP O/Ps
 ARE CORRECT. IF CORRECT, IT RETURNS
 TO PROGRAMME OTHERWISE CONTROL IS
 TRANSFERED TO 'FAULT'

| | | | | |
|------|--------|-------|-------|--|
| OADF | | CHCK1 | | |
| OADF | FEOA | CPI | OAH | } If chip O/Ps are not correct, chip is bad |
| OAE1 | C2140A | JNZ | FAULT | |
| OAE4 | C9 | RET | | Else, Return |

NAME OF SUBROUTINE CHCK2
 INPUTS
 OUTPUTS SAME AS CHCK1
 CALLS
 DESTROYS
 DESCRIPTION

| | | | | |
|------|--------|-------|-------|--|
| OAE5 | | CHCK2 | | |
| OAE5 | FEO5 | CPI | 05H | } If chip O/Ps are not correct, chip is bad |
| OAE7 | C2140A | JNZ | FAULT | |
| OAEA | C9 | RET | | Else, Return |

NAME OF SUBROUTINE T90
 INPUTS NONE
 OUTPUTS NONE
 CALLS LOAD, CHCK3

DESTROYS H, L, F/Fs

DESCRIPTION 'T90' TESTS IC 7490

| | | | | | |
|------|--------|-------|------|--------|---|
| OC00 | | T90 | | | |
| OC00 | F5 | | PUSH | PSW | Save status |
| OC01 | C5 | | PUSH | B | Save BC |
| OC02 | D5 | | PUSH | D | Save DE |
| OC03 | 3E90 | | MVI | A, 90H | } Load '7490 Test Programme' status |
| OC05 | 32FF0F | | STA | ICNO | |
| OC08 | 3E00 | | MVI | A, 00H | } Load repeat loop absent status |
| OC0A | 32FE0F | | STA | RPTLP | |
| OC0D | | T9005 | | | |
| OC0D | 3E07 | | MVI | A, 07H | } Make R _g (1) and R _g (2) HIGH and read chip O/Ps |
| OC0F | CD0DOA | | CALL | LOAD | |
| OC12 | FE09 | | CPI | 09H | } If the O/Ps are not correct The chip is bad |
| OC14 | C2140A | | JNZ | FAULT | |
| OC17 | 3E19 | | MVI | A, 19H | } Else make R ₀ (1) and R ₀ (2) HIGH and read chip O/Ps |
| OC19 | CD0DOA | | CALL | LOAD | |
| OC1C | CD580C | | CALL | CHCK3 | Check if O/Ps are correct |
| OC1F | 3E1B | | MVI | 1BH | } Again make R ₀ (1) and R ₀ (2) HIGH and read chip O/Ps |
| OC21 | CD0DOA | | CALL | LOAD | |
| OC24 | CD580C | | CALL | CHCK3 | Check if O/Ps are correct |
| OC27 | 3E09 | | MVI | 09H | } Disable asynchronous inputs and make I/PA=1 |
| OC29 | CD0DOA | | CALL | LOAD | |
| OC2C | CD580C | | CALL | CHCK3 | Check if O/Ps are correct |
| OC2F | 0601 | | MVI | B, 01H | Make B register for 'Correct O/Ps' |

| | | | | |
|------|--------|------|-------|---|
| OC31 | OE00 | MVI | C,00H | Make C counter for number of test inputs given |
| OC33 | | | | |
| OC33 | 3E0A | MVI | A,0AH |] Make I/PA=0 and check chip O/Ps |
| OC35 | CD0DOA | CALL | LOAD | |
| OC38 | B8 | CPM | B |] It O/Ps are not correct, chip is bad |
| OC39 | C2140A | JNZ | FAULT | |
| OC3C | 3E15 | MVI | A,15H |] Else, make I/PA=1 and check chip O/Ps |
| OC3E | CD0DOA | CALL | LOAD | |
| OC41 | B8 | CPM | B |] If O/Ps are not correct chip is bad |
| OC42 | C2140A | JNZ | FAULT | |
| OC45 | 04, | INR | B | Increment correct output register |
| OC46 | 0C | INR | C | Increment 'number of test inputs' given counter |
| OC47 | 79 | MOV | A,C |] Have all test inputs been given? |
| OC48 | FE09 | CPI | 09H | |
| OC4A | C2330C | JNZ | T9010 | No -go back to programme |
| OC4D | 3E08 | MVI | A,08H |] Yes - make I/PA=0 and read chip O/Ps |
| OC4F | CD0DOA | CALL | LOAD | |
| OC52 | CD580C | CALL | CHCK3 | Check if O/Ps are correct |
| OC55 | C3DB09 | JMP | GOOD | Chip is Good |

| | |
|--------------------|-----------------|
| NAME OF SUBROUTINE | CHCK3 |
| INPUTS |] SAME AS CHCK1 |
| OUTPUTS | |
| CALLS | |
| DESTROYS | |

DESCRIPTION

| | | | | |
|------|--------|-------|-------|--|
| OC58 | | CHCK3 | | |
| OC58 | FE00 | CPI | 00H |] If O/Ps are not correct chip is faulty |
| OC5A | C2140A | JNZ | FAULT | |
| OC5D | C9 | RET | | Else, return to programme |

NAME OF SUBROUTINE T93

INPUTS NONE
 OUTPUTS NONE
 CALLS LOAD, CHCK3

DESTROYS H,L,F/Fs

DESCRIPTION 'T93' TESTS IC 7493

| | | | | | |
|------|--------|-------|-------|---|--|
| 0C5E | | T93 | | | |
| 0C5E | F5 | PUSH | PSW | Save status | |
| 0C5F | C5 | PUSH | B | Save BC | |
| 0C60 | D5 | PUSH | D | Save DE | |
| 0C61 | 3E93 | MVI | A,93H |] Load '7493' Test programme status' | |
| 0C63 | 32FF0F | STA | ICNO | | |
| 0C66 | 3E00 | MVI | A,00H |] Load repeat loop absent status | |
| 0C68 | 32FE0F | STA | RPTLP | | |
| 0C6B | | T9305 | | | |
| 0C6B | 3E06 | MVI | A,06H |] Make R ₀ (1) and R ₀ (2) HIGH and read chip O/Ps | |
| 0C6D | CD0DOA | CALL | LOAD | | |
| 0C70 | CD580C | CALL | CHCK3 | Check if O/Ps are correct | |
| 0C73 | 3E05 | MVI | A,05H |] Make I/PA=1 and read chip O/Ps | |
| 0C75 | CD0DOA | CALL | LOAD | | |
| 0C78 | CD580C | CALL | CHCK3 | Check if O/Ps are correct | |
| 0C7B | 0601 | MVI | B,01H | Make B, the 'correct outputs' register | |
| 0C7D | 0E00 | MVI | C,00H | Make C the counter for number of test inputs given | |
| 0C7F | | T9310 | | | |
| 0C7F | 3E02 | MVI | A,02H |] Make I/PA=0 and read chip O/Ps | |
| 0C81 | CD0DOA | CALL | LOAD | | |
| 0C84 | B8 | CPM | B |] If chip O/Ps are not correct, chip is bad | |
| 0C85 | C2140A | JNZ | FAULT | | |
| 0C88 | 3E05 | MVI | A,05H |] Else make I/PA=1 and read chip O/Ps | |
| 0C8A | CD0DOA | CALL | LOAD | | |

| | | | | |
|------|--------|------|-------|---|
| OC8D | B8 | C.PM | B |] If chip O/Ps are not correct, chip is bad |
| OC8E | C2140A | JNZ | FAULT | |
| OC91 | 04 | INR | B | Increment 'Correct O/P' register |
| OC92 | 0C | INR | C | Increment 'number of test inputs given' counter |
| OC93 | 79 | MOV | A,C |] Have all test inputs been given? |
| OC94 | FE0F | CPI | 0FH | |
| OC96 | C27F0C | JNZ | T9310 | No - Go back to programme |
| OC99 | 3E02 | MVI | A,02H |] Yes - make I/PA=0 |
| OC9B | CD0DOA | CALL | LOAD | |
| OC9E | CD580C | CALL | CHCK3 | Check if chip O/Ps are correct |
| OCA1 | C3DB09 | JMP | GOOD | The chip is good |

PROGRAMME TABLES

| | | | | |
|-------|----------|-----|-------|-----------------------------|
| OFEO | GOODMG | | | Good Message |
| OFEO | ODOA492E | DB | | LF,CR,I,., |
| OFOE4 | 432E4953 | | | C,.,I,S, |
| OFE8 | 20474F4F | | | SP,G,O,O, |
| OFEC | 44ODOA | | | D,LF,CR |
| OOOF | IGOODM | EQU | 0FH | Length of GOODMG |
| OFFO | BADMG | | | 'BAD' Message |
| OFFO | ODOA492E | | | OF,CR,I,., |
| OFF4 | 432E4953 | | | C,.,I,S, |
| OFF8 | 20424144 | | | SP,B,A,D, |
| OFFC | ODOA | | | LF,CR |
| OOOE | LBADM | EQU | 0EH | Length of BADMG |
| OFFE | RPTLP | EQU | OFFEH | RPT LOOP INDICATOR LOCATION |
| OFFF | ICNO | EQU | OFFFH | IC NO INDICATOR LOCATION |

C H A P T E R - 6

UNIVERSAL IC TESTER

6.1 INTRODUCTION

The procedure for testing ICs in previous chapters requires different personality cards for each IC to be tested. This implies additional hardware for each additional IC testing programme to be included in the IC tester repertoire. Also, each time a different type of IC is to be tested, a new card has to be connected to the microprocessor ports. Obviously, for easy and efficient testing, a universal IC tester is required, which can test any IC using the same hardware, provided, of course, that the software for testing the IC exists in the IC tester memory.

6.2 REQUIREMENTS

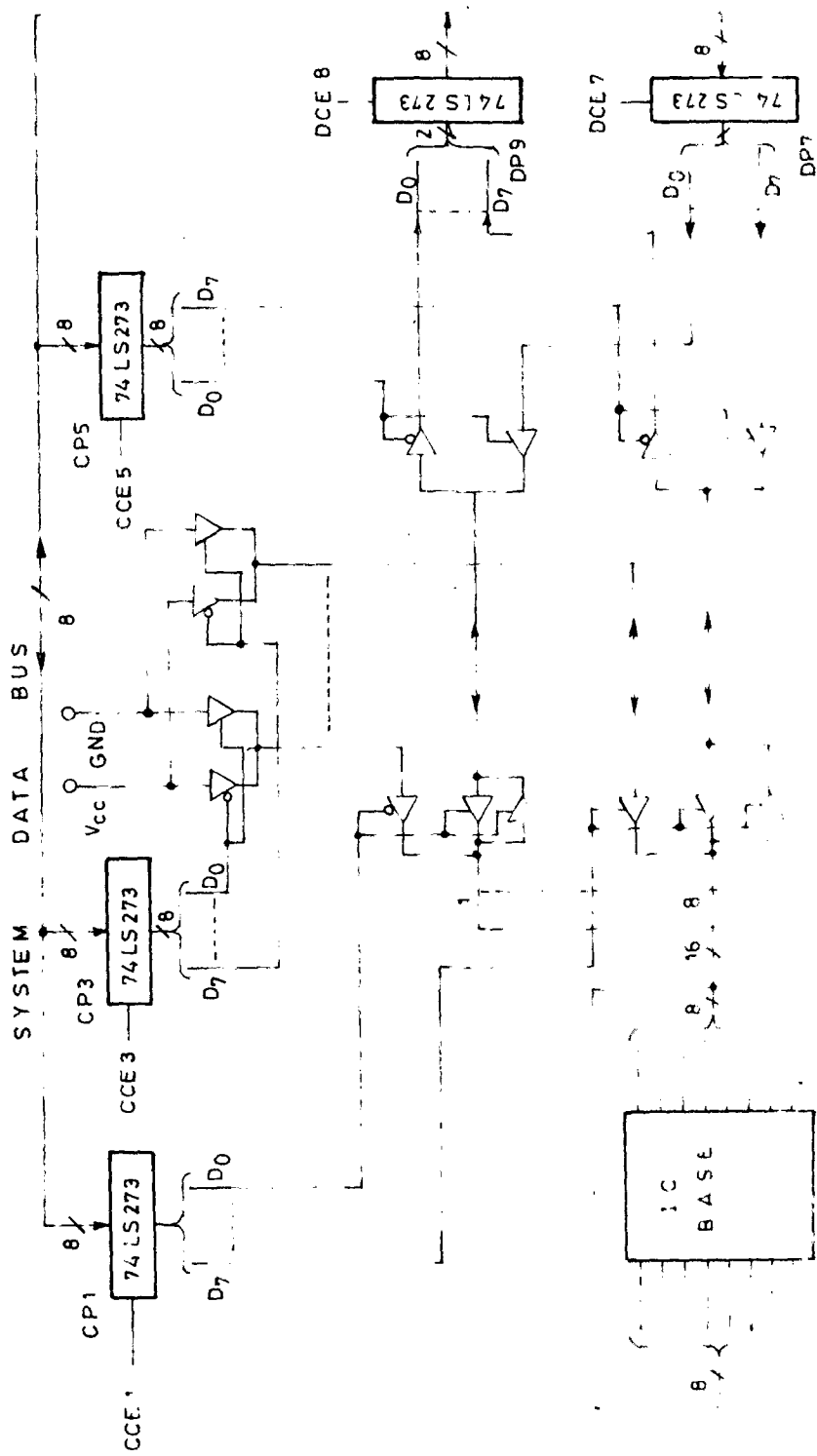
It is very seldom that two ICs having identical pin configuration (in respect of V_{CC} , Ground, Input and Output pins) will be encountered. Even ICs having identical logic functions but different parameters may, at times, have different pin configuration. The basic requirement of a universal IC tester is, therefore, that each pin of the IC under test should be programmable as an I/P, O/P, V_{CC} or Ground. Before the test inputs are impressed upon the IC, the test programme must be able to programme each pin to operate in the desired mode.

6.3 DESIGN OF UNIVERSAL IC TESTERS

Fig.6.1 describes the basic idea behind the design of a universal IC tester. Tri state buffers 74LS125 and 74LS126 have been used to programme different pins in the desired mode. Both these ICs have four tri-state buffers, each buffer having independent enable. The enable signals for 74LS125 are active Low and enable signals for 74LS126 are active HIGH. The pin configuration of these ICs is given in Appendix 'E'.

If the enable signal of a 74LS125 buffer is shorted with the enable signal of a 74LS126 buffer, the common line will then control selection of any one of the two buffers. A HIGH on this line will select the 74LS126 buffer, disabling the 74LS125 buffer. A Low on the line will select the 74LS125 buffer and disable the 74LS126 buffer.

As shown in Fig.6.1, each line joining a particular IC pin, can be controlled by different enable signals. The enable signals form different control ports. Since the IC pins should stay in a particular mode throughout the time of testing, there is a necessity of latching the control word at the beginning of the test programme. This is accomplished by using 74LS273 latch, which is a tri state 8 bit latch.



NOTE: CCE DENOTES CONTROL CHIP ENABLE.

DCE DENOTES DATA CHIP ENABLE.

FIG. 6.1 UNIVERSAL I.C. TESTER

6.4 PROGRAMMING THE UNIVERSAL IC TESTER

Diagnostic programmes written for various ICs in chapter 5 can be used with some modifications in universal IC tester. Depending upon the pin configuration of the IC under test, appropriate control words should be sent to control ports CP1 to CP6. This is the only modification required. A sample initialization programming for programming the pins of the universal IC tester is given in Appendix 'H', for IC 7400 chip. The latching arrangement provided will ensure that the pins remain programmed for the duration of test. Required test inputs can now be given to the IC chip and the O/Ps read into the microprocessor.

6.5 COMMENTS

The universal IC tester could not be fabricated due to non-availability of components. The list of components required is as under:

1. 74LS273 10 Nos
2. 74LS125 16 Nos
3. 74LS126 12 Nos

CHAPTER - 7CONCLUSION

7.1 SUMMARY OF THE WORK

In this thesis, a digital IC tester, operating through a CRT terminal, has been developed. This IC tester requires individual test cards for different ICs to be tested and test cards for IC 7400, 7476, 7490 and 7493 have been fabricated. Software for testing these ICs has also been developed.

The HIL-2961 to ADM-3A interface has been developed in complete details viz., power supply unit, baud rate generator, and 'RS232-C interface and 8251A USART interface' unit. Besides the baud rate of 4800 used presently, the baud rate generator can also generate many other baud rates upto 19200 to facilitate microprocessor interfacing with commonly used peripherals like CRT and TTY. The monitor developed for the IC tester is sufficient for executing various commands required for testing IC chips.

The Universal IC tester discussed in Chapter 6 obviates the need for different hardware for different IC chips and illustrates the flexibility and efficiency of microprocessor in programming different IC pins of the IC under test.

The successful working of various designed units has helped in consolidating the knowledge gained on micro-processor applications in the class-room. This has been the most significant gain of this work.

7.2 RECOMMENDED DEVELOPMENTS

Although the IC tester monitor is powerful enough to execute various commands for testing an IC chip, it does not have the capability of executing all the commands necessary for total control of the microprocessor. Other functional commands provided on the HIL-2961 keyboard and additional facilities like block move including appropriate change in branch instructions can be developed with a little more work. Complete monitor can then be stored in an EPROM to avoid loading it each time the supply is switched ON.

The universal IC tester can also be developed based upon the details provided in Chapter 6. Thereafter the IC tester will be able to test all the ICs having 14 or 16 pin configuration, depending upon the software development for testing various ICs.

REFERENCES

1. Hindustan Instruments Limited 'Microprocessor Trainer', HIL-2961 User's Manual'.
2. Lee, Samuel C. 'Digital Circuits and Logic Design', New Delhi : Prentice Hall of India Private Ltd. 1980.
3. Friedman, A.D. and Menon, P.R., 'Fault Detection in Digital Circuits', Englewood Cliffs : Prentice Hall International, INC.; 1971.
4. Fridrich, M. and Davis, W.A., 'Minimal Fault Tests for Combinational Networks', IEE Trans. Comput., Vol. C-23, Aug. 1974, pp 850-859.
5. Lear Siegler Inc. EID 'ADM-3A, Interactive Display Terminal, Operator's Handbook', 1975.
6. Garland, Harry 'Introduction to Microprocessor System Design', Tokyo : McGraw Hill Kogakusha, Ltd., 1979.
7. Rafiquzzaman, Mohamed 'Microcomputer Theory and Applications With the INTEL SDK-85', New York : John Wiley & Sons, Inc., 1982.
8. Smith, Lionel : application note AP-16, 'Using the 8251 Universal Synchronous/Asynchronous Receiver/Transmitter, Santa Clara : INTEL Corporation, 1976.
9. Tocci, Ronald J. 'Digital System : Principles and Applications', Englewood Cliffs : Prentice Hall INC., 1977.
10. INTEL, 'INTEL MCS-80 System Design Kit User's Manual', Santa Clara : INTEL Corporation.

11. INTEL, 'MCS-80/85TM Family User's Manual',
Santa Clara : INTEL Corporation,
October 1979.
12. Texas Instruments Incorporated, 'The TTL Data Book
for Design Engineers', Second Edition, 1976.
13. TOCCI, Ronald J. and Laskowski, Lester.
'Microprocessors and Microcomputers :
Hardware and Software', Englewood
Cliffs : Prentice Hall Inc., 1979.

.. ..

APPENDIX -- A

INTEL 8085 A : BRIEF DESCRIPTION (11)

A.1 SPECIAL FEATURES

8085 A is a single chip, 8 bit, N-channel central processing unit having following special features:

1. Single +5 V supply
2. 100% software compatibility with 8080 A
3. 1.3 μ s instruction cycle
4. On chip clock generator (with external crystal, LC or RC network)
5. On chip system controller; advanced cycle status information available for large system control
6. Four vectored interrupt inputs (one is non-maskable) plus an 8080 A compatible interrupt
7. Serial In/Serial out port
8. Decimal, binary and double precision arithmetic
9. Direct addressing capability to 64 K bytes of memory

A.2 FUNCTIONAL BLOCK DIAGRAM

Functional block diagram of internal architecture of 8085 A is shown in Fig.A.1

A.3 FUNCTIONAL PIN DEFINITION

The pin configuration of 8085 A is shown in Fig.A.2. Function of each pin is given below: