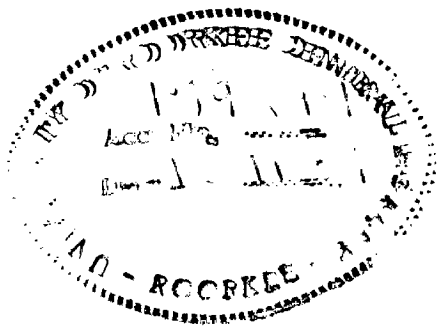
INTEL 8086 BASED CONTROLLER FOR CSTR CONTROL

A DISSERTATION

Submitted in partial fulfilment of the
requirements for the award of the Degree
of
MASTER OF ENGINEERING
in
ELECTRICAL ENGINEERING
(Systems Engineering and Operation Research)

By

RAKESH BHATNAGAR



DEPARTMENT OF ELECTRICAL ENGINEERING
UNIVERSITY OF ROORKEE
ROORKEE-247 667 (INDIA)
February, 1987

CANDIDATE'S DECLARATION

I hereby, certify that the work which is being presented in the dissertation entitled, 'INTEL 8086 BASED CONTROLLER FOR CSTR CONTROL' in partial fulfilment of the requirements for the award of the degree of MASTER OF ENGINEERING in ELECTRICAL ENGINEERING with specialization in SYSTEMS ENGINEERING AND OPERATION RESEARCH, submitted in the Electrical Engineering Department, University of Roorkee, Roorkee [India], is an authentic record of my own work carried out for a period of about six months from August, 1986 to February, 1987, under the supervision of Sh. M.K.Vasantha, Reader, Electrical Engineering Department, University of Roorkee and Sh. B.Mohanty, Lecturer, Chemical Engineering Department, University of Roorkee, Roorkee, India.

The matter embodied in this dissertation has not been submitted by me for the award of any other degree.

Dated 2/3/87

Rbhatnagar

[RAKESH BHATNAGAR]

This is to certify that the above statement made by the candidate is correct to the best of our knowledge.

MK Vasantha 28/2/87

[M.K.VASANTHA]
READER
ELECTRICAL ENGINEERING DEPARTMENT
UNIVERSITY OF ROORKEE
ROORKEE-247 667, INDIA

Bikash Mohanty 28/2/87

[B.MOHANTY]
LECTURER
CHEMICAL ENGINEERING DEPARTMENT
UNIVERSITY OF ROORKEE
ROORKEE-247 667, INDIA

ACKNOWLEDGEMENT

I wish to express my deep sense of gratitude to my guides, Sh. M.K.Vasantha, Reader, Electrical Engineering Department, University of Roorkee, Roorkee and Sh. B. Mohanty, Lecturer, Chemical Engineering Department, University of Roorkee, Roorkee, without whose co-operation, inspiration and guidance it would have been impossible to complete the thesis.

I would fail in my duty if I do not thank Mr. A.Garg, Proprieter, M/S Electrotech, Roorkee for helping me in getting various units of my thesis developed.

Thanks are due to Sh. N.Aterkar, M/S. Soilex Consultants Pvt.Ltd., Roorkee for having helped me out of number of difficulties.

I would like to greatfully acknowledge the direct or indirect part played by Prof. P.Mukhopaddhya (H.O.D.), Prof. A.K.Pant and Prof. H.K.Verma, Electrical Engineering Department, University of Roorkee, Roorkee in seeing through the thesis completed.

The completion of thesis with all the help available would not have been possible without the co-operation and advise of my close friends, Mr. A.Bhatia, Mr. G.C.Agnihotri, Mr. V.Pande, Mr. P.Garg and Mr. C.Raje, who have been associated with the thesis work right from the budding to the completion stage.

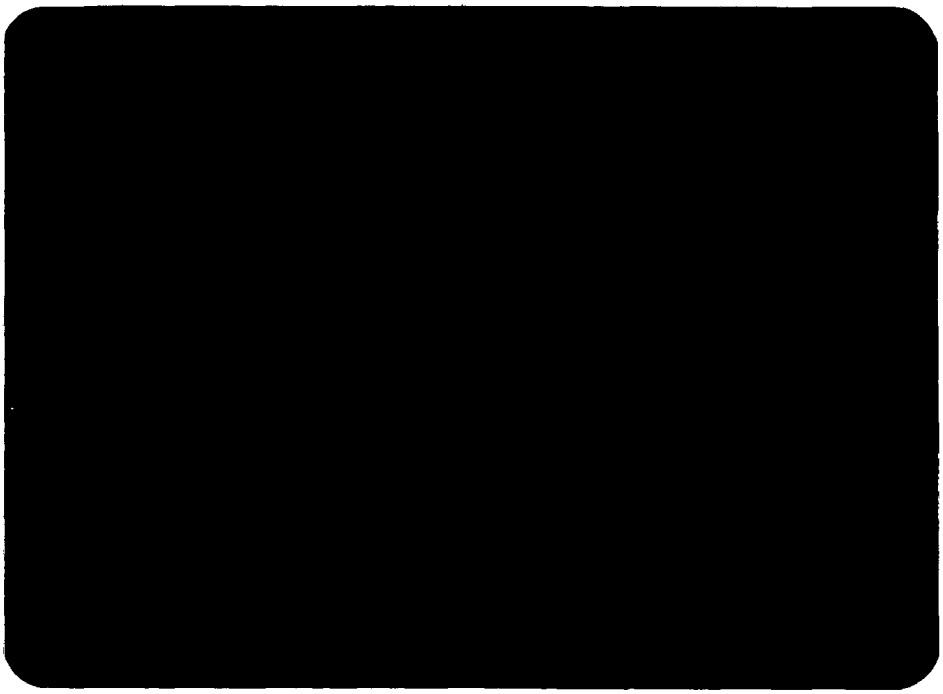
The up and Computer lab. staff, specially Mr. K.Singh, and Mr. R.Singh, who have played a role of prime importance in getting my thesis completed, also deserve a special word of thanks from me.

Thanks are also due to Mr. D.C.Bhardwaj, E and C.Engg. Department for his effecient and time bound typing of the thesis.

In the end, I am greatful to all whose name I have missed and who have played a part in seeing through the thesis to the final phase.

RBhatnagar

RAKESH BHATNAGAR



1) SIGN-ON MESSAGE ON CRT

ABSTRACT

In the present dissertation, an effort has been made to control the temperature and level of continuously flowing water in a Continuous Stirred Tank Reactor (CSTR). A CSTR and a digital controller using 8086 μp have been developed for implementing a Proportional-Integral-Derivative (PID) control scheme, through software, to control the temperature. A level transducer, using capacitance-to-frequency conversion principle and a valve, coupled with a stepper motor, for controlling the level have also been developed.

For controlling the temperature, a 1-ϕ half-controlled SCR bridge has been fabricated. The firing circuit required for the bridge and an A/D converter module have also been developed for the above purpose. A precise control of temperature in the range of $\pm$ 0.25^{o}C has been observed.

A floating point arithmetic has been developed which can be used for accurate calculations by PID control scheme. Various other routines, necessary for level control, have also been developed, but these are not used for the present work.

The man- m/c communication has been achieved through a CRT. Some functional commands have been developed for this purpose.

CONTENTS

contd...

CHAPTER - I

INTRODUCTION

In Chemical Industries, Continuous Stirred Tank Reactor (CSTR) with heating system plays a major role in carrying out general purpose reactions at elevated temperatures. The controlling factors that effect the performance of a CSTR include level of the fluid in the tank, flow rate at the inlet and outlet of the tank and temperature of the fluid in the tank. Out of all these temperature is the most important parameter as it controls the reaction rate of a raction. Thus it should be controlled with a greater accuracy. The efficiency of CSTR, an important building block of Chemical Industries can be improved to a greater degree by on-line controlling the input and output variables. This, in turn, will help in maintaining the product quality and reducing the production cost.

There are several control schemes which have been developed for the process control. For the present work, PID control scheme has been used. In the past days, analog controllers were very popular for the process control. Analog controllers consist of mainly the operational amplifiers which are used for building multiplier, integrator, differentiator and summer blocks for the PID scheme. But now a days, digital controllers, based on Microprocessors are successfully replacing the analog controllers. This yields certain advantages, as summarised below:

1) A digital controller is usually cheaper than its analog counterpart.

2) It offers a significant flexibility by changing the control parameters or even the control algorithm because only the software is required to be changed.

3) It is smaller and lighter than its analog counterpart and the power consumption is also very low.

4) It offers great reliability since no. of components in the system noise are successfully overcome.

5) The importance of up based control systems becomes more distinct when more no. of variables and control loops are dealt with.

All the considerations as mentioned above led to the present investigation with following objectives:

1) To study the VMC-86/3 Microcomputer kit, m.fd by Vinytics Peripherals Pvt.Ltd., Ghaziabad, along with 8259 interrupt controller for the control purpose.

2) To develop a fast digital controller with the help of a 8086 up.

3) To develop support devices like A/D converter, SCR and based single phase converter, level transducer stepper motor controlled control value for level control.

The photographs of the complete system and individual modules along with the C-F converter and SCR firing circuit waveforms are added after the last chapter.

4) To develop software for support devices, floating point arithmetic, PID algorithm and serial communication through CRT.

5) To test the working of above hardware and software developments by controlling temperature of a CSTR.

The report of the work done in this dissertation is divided into eight chapters. After having given a brief introduction of objectives of the work to be done, i.e., control of CSTR, using PID control scheme in Chapter-I, a detailed description of the system is included in Chapter- II. Chapter -III deals with the description of temperature transducer and design and fabrication of level transducer. The development of SCR firing circuit and A/D converter card form an integral part of the chapter -IV, which ends with a description of stepper motor driver card, m.fd by Vinytics Peripherals Pvt.Ltd., Ghaziabad. A method of estimation of PID control parameters is given in Chapter -V, which also includes the description of various control schemes and the software implementation of PID control scheme. Software development along with flow charts and program listing is described in Chapter VI. The results obtained from the experimentation conducted on the practical system are discussed in Chapter VII, and are reported in Appendix-A. Conclusions and recommendations for further developments emerged out of the present work are given in Chapter - VIII.

CHAPTER II

SYSTEM DESCRIPTION

This chapter describes the experimental set up for the process to be controlled and the details of micro - computer kit used to control the process.

2.1 EXPERIMENTAL SET-UP

The experimental set up has been depicted schematically in fig. 2.1-1, as shown in photograph no.2 It consists of an overhead tank (1) having a 4 inch overflow line (2) to maintain constant head, a water outlet (13), regulating values (3,4), a solenoid valve (5), mechanical stirrer (6), CSTR (7), two number of main heaters (8) with 1.3 KW capacity each, an auxiliary heater (9) with a capacity of 1.3 KW, temperature senser (copper constantan thermocouple, m.f.d M/S Thermoelectrics, Neitherlands)(10), capacitive level senser (11) and stepper motor controlled value (12).

2.1.1 CSTR

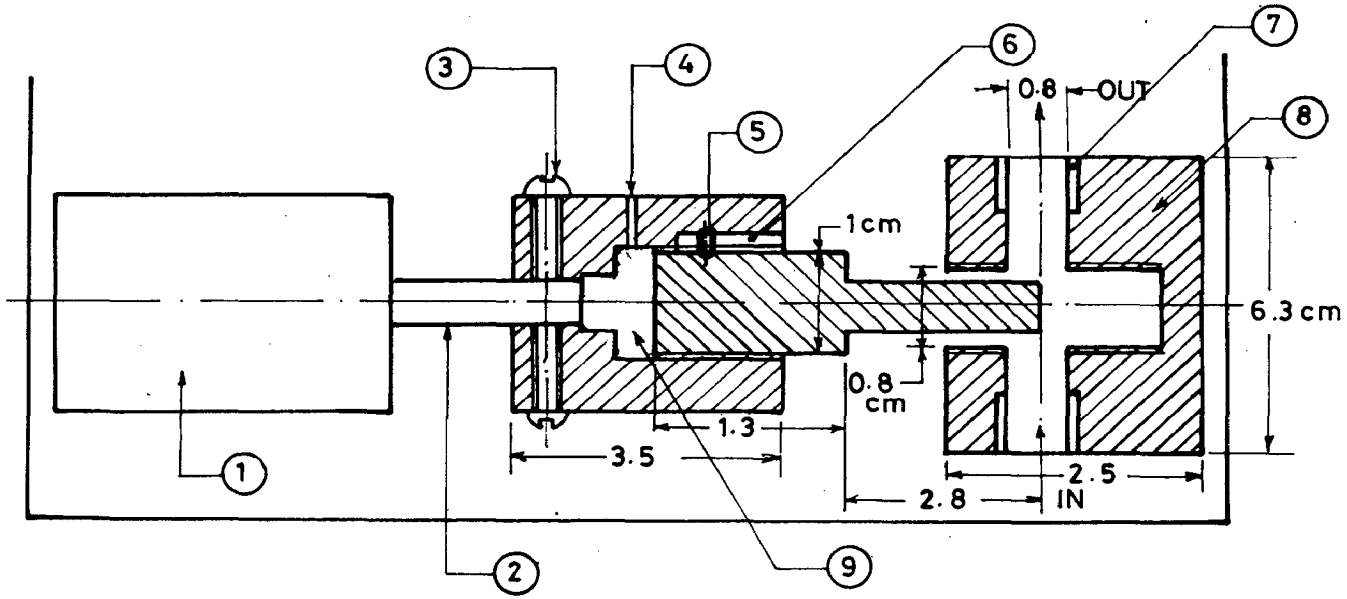
Continuous Stirred Tank heater (CSTR) is a vertical shell as shown in fig. 2.1-1 . The I.D. and height of the cylinder are 21 cm. and 50 cm. respectively. The CSTR is fitted with a mechanical stirres (6) to homogenize the liquid continuously by stirring it. A copper constantan thermocouple (10) is used to measure the temperature of

CSTR. A specially developed capacitive level transducer with a precision C-F converter is used to monitor the liquid level inside the CSTR. A 4-KW heater with three immersion heating elements, each having 1.33 KW capacity is used for heating the contents of CSTR. Out of these 3 heaters, 2 heaters are used to deliver 80 percent of the energy required. The third heater is used through a µp controlled SCR converter bridge (sec. 4.1) for control purpose.

Fig. 2.1-1 shows the relative positions of inlet pipe, thermocouple, level senser, heaters, stirre. inside the CSTR. The outlet of the CSTR is connected with a stepper motor controlled value. The outlet flow rate of the CSTR can be controlled by this value directly by µp. The level of the CSTR can be controlled by controlling the outflow.

2.1.2 OVERHEAD TANK

It consists of a rectangular mild steel tank, having a 4-inch overflow line, draining the water if its level in the tank exceeds a given height. The tank is painted with primer followed by an enamel paint. This eliminated any possibility of the tank getting rusted and therby making the valves choke and sensors dirty. A strainer is also provided just at the month of the exit pipe to prevent any slush getting into the system.



1. STEPPER MOTOR
2. SPINDLE OF MOTOR
3. SPINDLE FIXING SCREW
4. AIR VENT
5. STUD TO LOCK ROTARY MOTION
6. KEY HOLE FOR STUD MOVEMENT
7. FLOW CONTROL SCREW
8. VALVE ASSEMBLY
9. MATCHING HOLE FOR THE LEFT END OF SCREW

FIG. 2.1-3 CONTROL VALVE WITH STEPPER MOTOR

2.1.3 STEPPER MOTOR CONTROLLED VALVE

Fig.2.1-3 shows the details of the control valve developed for the present work. The control valve is operated with the help of a stepper motor which precisely controls the linear movement of the flow control screw (7). The stepper motor is activated through the up and the pulse trains are issued to rotate it upto a predetermined angle. The rotary motion thus created is transferred to the flow control screw through (5,6,9) which in turn moves in or out in the flow area to increase or decrease resistance in the flow path as required. As amount of flow is equal to head across the valve divided by resistance, the movement of screw controls the flow rate.

The present valve is designed to operate at very low pressure and hence no sealing technique is used to make it leak proof. Under present circumstances the valve works excellently with very little leakage.

2.2 SALIENT FEATURES OF VMC-86/3

VMC-86/3 is a single board Microprocessor Training/ Development Kit manufactured by the company-Vinytics and is configured around the Intel's 16 bit Microprocessor 8086 in maximum mode. The kit can be used for process control and to develop software for the 8086 systems. The co-processor 8087 and I/O processor 8089 can also be added on the board. 8086 CPU can also be replaced by 8088 CPU.

It can be communicated with the kit through a key - board having 28 keys and eight seven segment displays. The kit also has the capacity of interacting with teletypewriter, CRT terminal and Audio Cassette Recorder.

2.2.1 SYSTEM SPECIFICATIONS

PROCESSORS:

Central Processor	: 8086, 16 bit up operating in maximum mode or 8088, 8 bit up. Both processors can be operated at a max. clock frequency of 15 MHz.
Co-Processor	: 8087 Numeric data processor
I/O Processor	: 8089 I/O Processor

MEMORY:

EPROM	: 32 K bytes loaded with monitor, expandable to 128 K bytes using four 27256.
RAM	: 16 K bytes of CMOS RAM expandable to 128 K bytes using sixteen 6264.

INPUT/OUTPUT:

Parallel	: 72 lines using three 8255.
Serial	: 1) 20 mA Current Loop 2) EIA RS-232-C (Main) 3) EIA RS-232-C (Aux.) 4) Audio Cassette Recorder Interface.

Interrupt (256 Vectored)	:	8 different level interrupts through 8259 A
Timer/Counter	:	Three 16 bit Timer/Counter through 8253.
Other Interfaces	:	EPROM Programmer for 2732/2732A/ 2764/27128/27256 using specific personality module for each of them.
Keyboard and Display	:	28 keys and 8 seven segment display.
Bus	:	All address, data and control signals (TTL compatible) available at edge connecter as per Multibus. The kit has its own Resident Bus.
Power Supply	:	5V ± 5 % , 2.5 A for kit and ± 12V, 400 mA for TTY/CRT.

2.2.2 HARDWARE DESCRIPTION

1) CPU (8086 A-4) :

8086 is a 16 bit μp having 16 data lines and 20 address lines. The lower 16 address lines are multiplexed with 16 data lines. The address lines are latched by using 74LS 373 as shown in Appendix D. The 8086 is used in maximum mode (MN/ $\overline{MX}$ input held logically low). The processor is designed to operate with the 8089 I/O processor and other processors in multiprocessing and distributed processing systems.

The INTR and TEST inputs to 8086 are pulled low. TEST is used by the coprocessor.

2) CO-PROCESSOR 8087 :

An 8087 Co-processor obtains its instructions from another processor, called a host. The co-processor monitors instructions fetched by the host and recognizes certain of these as its own, and executes them . A co-processor, in effect, extends the instruction set of its host computer.

3) I/O PROCESSOR 8089 :

The 8086 is designed to be used with the 8089 in high performance I/O applications. The 8089 conceptually resembles a up with two DMA channels and an instruction set specifically tailored for I/O operations, Unlike simple DMA controllers, the 8089 can service I/O devices directly, removing this task from the CPU. In addition, it can transfer data on its own bus or on the system bus, can match 8-bit or 16-bit peripherals to 8-bit or 16-bit buses, and can transfer data from memory to memory or from I/O device to I/O device.

4) CLOCK GENERATOR (8284)

The clock generator circuit is an Intel 8284 clock generator/driver. The circuit requires a crystal input which operates at a fundamental frequency of 14.7456 MHz. The clock generator/driver divides the crystal frequency by three to produce a 4.95 MHz CLK signal. Additionally, it gives a divide by two output of the

4.95 MHz CLK and is called PCLK (peripheral Clock), which is used by the various peripherals on the board. The system can operate either at 4.95 MHz or at 2.45 MHz. This is selected by a jumper connection on the board as shown below:

C ○ 4.9 MHz

B ○ CLK

A ○ 2.45 MHz.

The VMC 86/3 is supplied in 2.45 MHz configuration.

The clock generator/ driver provides two control signal outputs which are synchronized internally; RDY (ready) and RST (reset). RST is used to reset the VMC-86/3 to an initialized state and occurs when the $\overline{\text{RES}}$ input goes low (when power first is applied or when the system RESET key is pressed). The RDY2 input is made high and $\overline{\text{AEN2}}$ input low so as to make READY output high.

5) BUS CONTROLLER 8288) :

The 8288 is a Bus Controller which decodes status signals outputted by an 8089 or an 8086 in maximum mode. When these signals indicate that the processor is to run a bus cycle, the 8288 issues a bus command that identifies the bus cycle as memory read, memory write, I/O read, I/O write, etc. It also provides a signal that strobes the address into latches. The decoding is as follows:

$\overline{S}_2$	$\overline{S}_1$	$\overline{S}_0$	Processor State	8288 Command
0	0	0	Interrupt Acknowledge	$\overline{INTA}$
0	0	1	Read I/O Port	$\overline{IORC}$
0	1	0	Write I/O Port	$\overline{IOWC}$
0	1	1	Halt	None
1	0	0	Code Access	$\overline{MRDC}$
1	0	1	Read Memory	$\overline{MRDC}$
1	1	0	Write Memory	$\overline{MWTC}$, $\overline{AMWC}$
1	1	1	Passive	None

6) BUS ARBITER (8289) :

The 8289 is a Bus Arbitor that controls the access of a processor to multimaster system resources (typically memory) that is shared by two or more microprocessors (masters). Arbiters for each master may use one of the several priority resolving techniques to ensure that only one master drives the shared bus.

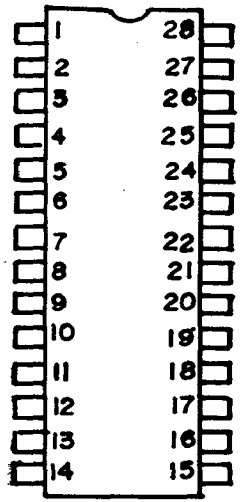
7) MEMORY :

The kit provides 32 K bytes of EPROM loaded with monitor and 16K bytes of CMOS RAM. The total onboard memory can be configured as follows:

EPROM - 16 K bytes using four 2732
32 K bytes using four 2764
64 K bytes using four 27128
128 K bytes using four 27256.

RAM -128 K bytes using sixteen 6264.

27256	27128	2764	2732A
V_{pp}	V_{pp}	V_{pp}	
A_{12}	A_{12}	A_{12}	
A_7	A_7	A_7	A_7
A_6	A_6	A_6	A_6
A_5	A_5	A_5	A_5
A_4	A_4	A_4	A_4
A_3	A_3	A_3	A_3
A_2	A_2	A_2	A_2
A_1	A_1	A_1	A_1
A_0	A_0	A_0	A_0
O_0	O_0	O_0	O_0
O_1	O_1	O_1	O_1
O_2	O_2	O_2	O_2
Gnd.	Gnd.	Gnd.	Gnd.



2732A	2764	27128	27256
	V_{cc} / $\overline{PGM}$	V_{cc} / $\overline{PGM}$	V_{cc}
			A_{14}
V_{cc}	N.C.	A_{13}	A_{13}
A_8	A_8	A_8	A_8
A_9	A_9	A_9	A_9
A_{11}	A_{11}	A_{11}	A_{11}
$\overline{OE}/V_{pp}$	$\overline{OE}$	$\overline{OE}$	$\overline{OE}$
A_{10}	A_{10}	A_{10}	A_{10}
$\overline{CE}$	$\overline{CE}$	$\overline{CE}$	$\overline{CE}$
O_7	O_7	O_7	O_7
O_6	O_6	O_6	O_6
O_5	O_5	O_5	O_5
O_4	O_4	O_4	O_4
O_3	O_3	O_3	O_3

FIG. 2.2-1

The system provides four 28 pin sockets for the EPROM area named as EPROM 0 to EPROM 3 and sixteen 28 pin sockets for the RAM area named as RAM0 to RAM 15. EPROM 0 to EPROM 3 can be defined to have either of 2732/2764/27128/27256 and the RAM 0 to RAM 15 is defined for 6264. The selection is done by changing the jumper connections in the Block Box, for the EPROMs.

SELECTION OF MEMORY CHIPS:

The RAM area, i.e. RAM 0 to RAM 15 are defined for 6264 CMOS RAM. But the ROM area, i.e., EPROM 0 to EPROM 3 can be defined to have 2732/2764/27128/27256. All the signal points of the 28 pin sockets except pin no. 26 and 27 are same for a 2732, 2764, 27128 and 27256 chips, as shown in Fig. 2.2.1. These signal points are named as T26 and T27 in the circuit diagram shown in Appendix D. For selecting either of the above mentioned EPROMs, jumper selection is to be done for T26, T27, JM1, JM2 and ADCD, as shown in the circuuit diagram.

The following table shows the jumper position for the different EPROMs.

	27256	27128	2764	2732
T27	A_{15}	V_{cc}	V_{cc}	V_{cc}
T26	A_{14}	A_{14}	V_{cc}	V_{cc}
JM2	-	A_{16}	A_{16}	A_{16}
JM1	-	-	A_{15}	A_{15}
ADCD	A_{16}	A_{15}	A_{14}	A_{14}

The EPROM sockets are named as EPROM 0 to EPROM 3. Socket 0 and Socket 1 are corresponding to the even bytes and socket 2 and socket 3 are corresponding to odd bytes. The monitor lies in sockets 1 and 3 in EPROM 27128.

The memory mapping of the various chips corresponding to sockets 0 to 3 is as follows:

	27256	27128	2764	2732
EPROM 0	E0000 to FFFFE	F0000 to F7FFE	F8000 to FBFFE	FC000 to FDFFE
EPROM 1	F0000 to FFFFE	F8000 to FFFFE	FC000 to FFFFE	FE000 to FFFFE
EPROM 2	E0001 to EFFFF	F0001 to F7FFF	F8001 to FBFFF	FC001 to FDFFF
EPROM 3	F0001 to FFFFF	F8001 to FFFFF	FC001 to FFFFF	FE001 to FFFFF

Thus, using 27128 in sockets 1 and 3 refers to an address F8000-FFFFF. Additionally, 2764 used in sockets 2 and 4 refers to the address F0000-F3FFFor F4000-F7FFF.

The memory sockets RAM 0 to RAM 7 are corresponding to even bytes and RAM 8 to RAM 15 are corresponding to odd bytes. The memory mapping of these are given below:

RAM 0 : 00000-03FFF	RAM 8 : 00001 - 03FFF
RAM 1 : 04000-07FFE	RAM 9 : 04001 - 07FFF
RAM 2 : 08000-0BFFE	RAM 10: 08001 - 0BFFF
RAM 3 : 0C000-0FFFE	RAM 11: 0C001 - 0FFFF
RAM 4 : 10000-13FFE	RAM 12: 10001 - 13FFF
RAM 5 : 14000-17FFE	RAM 13: 14001 - 17FFF
RAM 6 : 18000-1BFFE	RAM 14: 18001 - 1BFFF
RAM 7 : 1C000-1FFFE	RAM 15: 1C001 - 1FFFF

Thus 6264 having used in sockets 0 and 8 refers to the memory area 00000-03FFF.

8) I/O DEVICES :

(a) 8279

8279 is a general purpose programmable keyboard and display I/O interface device designed for use with the 8086 µp. It provides a scanned interface to 28 contact key matrix and scanned displays. It has got 16 x 8 display RAM which can be loaded or interrogated by the CPU. When a key is pressed, its corresponding code is entered in the FIFO queue of 8279 and is read by the µp. 8279 also refreshes the display RAM.

(b) 8255 :

8255 is a programmable peripheral interface (PPI) which basically acts as a general purpose I/O component

to interface peripheral equipments to the system bus. It has got three input/output ports of 8 lines each (Port A, Port B and Port C). Port C can be divided into two ports of 4 lines each named as Port C upper and Port C lower. Any input/output combination of port A, port B, port C upper and port C lower can be defined using the appropriate software commands. Nine I/O ports are provided by using three 8255 chips. The ports lines are brought out at connectors J1, J2, and J 3, the details of which are given in Appendix B. A word may be inputted or outputted using the same ports of 8255-I and 8255-II simultaneously.

(c) 8253 :

This chip is a programmable interval timer/counter and can be used for the generation of accurate time delays under software control. This chip has got three independent 16-bit counters each having a count rate of upto 2 MHz. The CLK, GATE and OUT signals of these timers are brought out at the J5 connector.

(d) 8251 :

This chip is a programmable communication interface and is used as a peripheral device. This device accepts data characters from the CPU in parallel form and then converts them into a continuous serial data stream for transmission. Simultaneously, it can receive serial data stream and converts them into

parallel data characters for the CPU. This chip will signal the CPU whenever it can accept a new character for transmission or whenever it has received a character for the CPU. The CPU can read the complete status of it at any time. In the kit, 8251 has been utilized for Main/Aux. RS-232-C interface and 20 mA current loop (i.e., for CRT/TTY interface).

(e) 8259 :

The 8259 is a device specifically designed for use in real time, interrupt driven μC systems. It manages eight levels of requests and has built in features for expandability to other 8259's. It is programmed by system's software as an I/O peripheral. A selection of priority modes is available to the programmer, which can be changed or reconfigured dynamically at any time during the main program. The IR0-IR2 lines of 8259 have been used by the system itself for 8087 co-processor and 8089 I/O processor. IR3-IR7 lines have been buffered and are brought out on the multibus. The details of multibus are given in Appendix B.

9) DISPLAY :

The kit provides eight digits of seven segment display; four digits for displaying address of any location or name of any register, and the rest four for the contents of the memory location or register. All the eight digits of the display are in hexadecimal notation.

10) BUFFERS :

The address, data and control lines are buffered by using the ICs 74-LS-245 and 74-LS-240 and are made available to the user at the PCB edge connector in the MULTIBUS configuration. All these lines have been made bi-directional.

11) PORT ADDRESSES:

The port addresses of the various I/O devices used in VMC-86/3 kit are given below:

DEVICE NAME	PORT NAME	PORT ADDRESS
8255-I	Port A	FFF8
	Port B	FFFA
	Port C	FFFC
	Control Word	FFFE
8255-II	Port A	FFF9
	Port B	FFFB
	Port C	FFFD
	Control Word	FFFF
8255-III	Port A	FFE0
	Port B	FFE2
	Port C	FFE4
	Control Word	FFE6
8279	Data Word	FFE8 or FFEC
	Command Word	FFEA or FFEE
8253	Counter 0	FFD8
	Counter 1	FFDA
	Counter 2	FFDC
	Control Word	FFDE

contd...

DEVICE NAME	PORT NAME	PORT ADDRESS
8251 (Main)	Data Word	FFF0 or FFF4
	Command Word	FFF2 or FFF6
8251 (Aux.)	Data Word	FFD0 or FFD4
	Command Word	FFD2 or FFD6
8259	Data word	FFC8 or FFCC
	Command Word	FFCA or FFCE
8089	Channel 1	FFC0
	Channel 2	FFC1

2.1.3 EXPANSION THROUGH MULTIBUS

The multibus is a general purpose multiprocessing system bus. All the designs based on VMC-86/3 are to be developed around this standard bus.

The address, data and control lines of 8086 in VMC-86/3 are buffered and are made available at the PCB edge connector, as per the MULTIBUS STANDARD.The pin assignment of MULTIBUS is given in Appendix B.

The multibus signal lines are divided into following sections:

1. Initialization Signal Line
2. Address and Inhibit Lines
3. Bus Contention Resolution Lines
4. Information Transfer Protocol Lines
5. Asynchronous Interrupt Lines
6. Power Supply Lines.

These are described below:

1. INITIALIZATION SIGNAL LINE

$\overline{\text{INIT}}$: The initialization signal resets the entire system to a predetermined state. $\overline{\text{INIT}}$ may be supplied by one of the bus masters or by external logic.

2. ADDRESS AND INHIBIT LINES:

(a) $\overline{\text{MA 0}}$ - $\overline{\text{MA 19}}$: 20 address lines are used to transmit the address of a memory location or an I/O port to be accessed.

(b) $\overline{\text{BHEN}}$: The least significant address line ADR0 and BHEN are used to specify that the 8-bit data will be transferred on the higher order 8 data lines or lower order 8 data lines. If the data transfer is 16bits, their combination enables data on all the 16 lines.

3. DATA LINES

$\overline{\text{SD0}}$ -$\overline{\text{SD15}}$: The 16 bidirectional data lines are used to exchange information with a memory location or I/O port. In 8 bit systems, only the lines $\overline{\text{SD0}}$ - $\overline{\text{SD 7}}$ are used.

4. BUS CONTENTION RESOLUTION LINES

(a) $\overline{\text{BCLK}}$: The negative edge of the Bus clock is used to synchronize bus contention. $\overline{\text{BCLK}}$ is asynchronous with the CPU clock. $\overline{\text{BCLK}}$ may be slowed, stopped or single stepped during debugging.

(b) $\overline{\text{CCLK}}$: The constant clock provides a clock signal of constant unspecified frequency.

(c) $\overline{\text{BPRN}}$: The Bus Priority In signal tells a bus master that no higher priority device is requesting use of the system bus. $\overline{\text{BPRN}}$ is synchronized with $\overline{\text{BCLK}}$. This signal is 'daisy chained' if serial priority arbiteration is used. When parallel priority arbitera-tion is used, a bus master generates $\overline{\text{BPRN}}$.

(d) $\overline{\text{BPRO}}$: This is Bus Priority Out signal . Like $\overline{\text{BPRN}}$, $\overline{\text{BPRO}}$ is daisy chained when serial peiority arbiteration is used, $\overline{\text{BPRO}}$ is fed to the $\overline{\text{BPRN}}$ input of the next priority module. When using parallel prio-rity arbiteration, a bus arbiter must provide this signal. $\overline{\text{BRPO}}$ is synchronized with $\overline{\text{BCLK}}$.

(e) $\overline{\text{BUSY}}$: The Bus Busy signal is applied by the current bus master to indicate that the system bus is in use. $\overline{\text{BUSY}}$ is used by other devices to determine whether or not they may acquire control of system bus. $\overline{\text{BUSY}}$ is synchronized with $\overline{\text{BCLK}}$.

(f) $\overline{\text{BREQ}}$: The Bus Request signal is used by devices to indicate that they wish to become bus master. $\overline{\text{BREQ}}$ is synchronized with $\overline{\text{BCLK}}$; it is not bussed on the mother board.

(g) $\overline{\text{CBRQ}}$: $\overline{\text{CBRQ}}$ is used by all potential bus masters to inform the current bus master that another master wishes to use the bus. If high, the current bus master

knows that no other device is requesting the bus, and therefore the present bus master is to retain the bus.

5. INFORMATION TRANSFER PROTOCOL LINES

A bus master that has control of the systm bus generates all data transfer control signals. All address signals (and data signal, when a write is to occur) must be stable at least 50 ns prior to the transfer control signal pulse and must remain valid for at least 50 ns after control signal pulse is removed.

Information transfer protocol lines are not synchronous with $\overline{\text{BCLK}}$.

(a) $\overline{\text{SMRD}}$: The memory Read Control indicates that the address of a Memory location has been placed on the address lines and that the contents of the address location are to be placed on data lines.

(b) $\overline{\text{SMWR}}$: The Memory Write control indicates that the address of the memory location has been placed on the address lines and the data to be written into the addressed memory location is going to be placed on the system data bus.

(c) $\overline{\text{SIRD}}$: The I/O Read Control indicates that the address of an input port has been placed on the system address lines and the data at that input port is to be placed on the-data lines.

(d) $\overline{\text{SIWR}}$: The I/O write control indicates that the address of an O/P port has been placed on the system address lines and the data is going to be outputted to that port.

(e) $\overline{\text{XACK}}$: The Transfer Acknowledge signal is used in handshaking by the selected bus slave to acknowledge to the bus master in response to the transfer control signal.

6. ASYNCHRONOUS INTERRUPT LINES

(a) $\overline{\text{INT3}}$ - $\overline{\text{INT7}}$: These priority interrupt request lines are used with parallel interrupt resolution circuitary.

(b) $\overline{\text{INTA}}$: The interrupt acknowledge signal is used by a bus master to acknowledge the interrupt signal placed by the external logic.

7. POWER SUPPLY LINES

Various regulated power supply lines are provided on the multibus. These include GND, +5V, -5V, +12V and -12V.

The circuit dig in Appendix D shows how these signals are brought at the PCB edge connector. The pin assignment of bus signals on MULTIBUS Board connector is given in Appendix B.

2.3 SYSTEM ASSEMBLY

The µp, process, instruments and transducers are assembled as shown in Fig. 2.3.1.

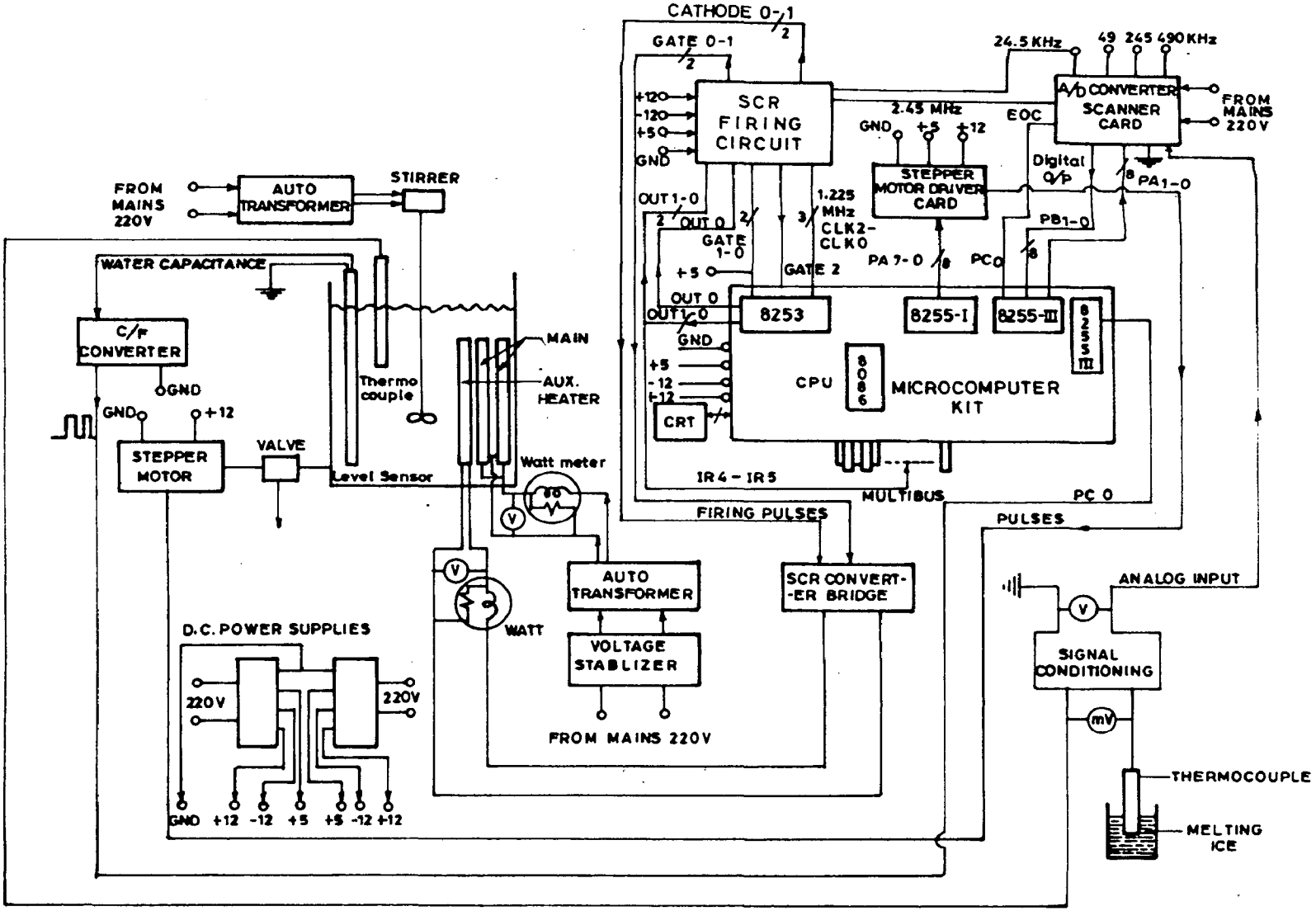


FIG. 2.3-1 SYSTEM ASSEMBLY

CHAPTER-III

TRANSDUCERS AND SIGNAL CONDITIONING

A transducer converts the variable to be monitored into a suitable electrical signal that may be an equivalent voltage, frequency etc. This O/P signal is suitably conditioned before it is fed to µp.

Temperature and level transducers used for the present work, with signal conditioning are discussed in this chapter.

3.1 TEMPERATURE TRANSDUCER AND SIGNAL CONDITIONING

A Copper Constantan Thermocouple, manufactured by M/S Thermoelectrics, Netherlands is used. Two thermocouples are used to measure the temperature of the CSTR. One of the thermocouples is put in cold junction (melting ice) and other at the CSTR as hot junction to mreaure the actual temperature of process fluid.

As the temperature -emf relationship of the thermocouple is non-linear, it is divided into six linear zones for the temperature range 0° to $70^{\circ}C$. The temperature-emf relationships for these zones are given below:

$$\text{Temp.} = 27.613402 \times \text{emf.} + 0.759278, \quad \text{for } 1.188 \geq \text{emf } (\mu v)$$

$$= 24.27180 \times \text{emf.} + 1.165120, \quad \text{for } 1.188 < \text{emf} \leq 1.600$$

$$= 23.809523 \times \text{emf.} + 1.904764, \quad \text{for } 1.600 < \text{emf} \leq 2.020$$

$$= 23.364486 \times \text{emf.} + 2.803738, \quad \text{for } 2.020 < \text{emf} \leq 2.448$$

$$= 22.831050 \times \text{emf.} + 4.109589, \quad \text{for } 2.448 < \text{emf} \leq 2.886$$

$$= 22.624434 \times \text{emf.} + 4.70542, \quad \text{for } 2.886 < \text{emf} \leq 3.328$$

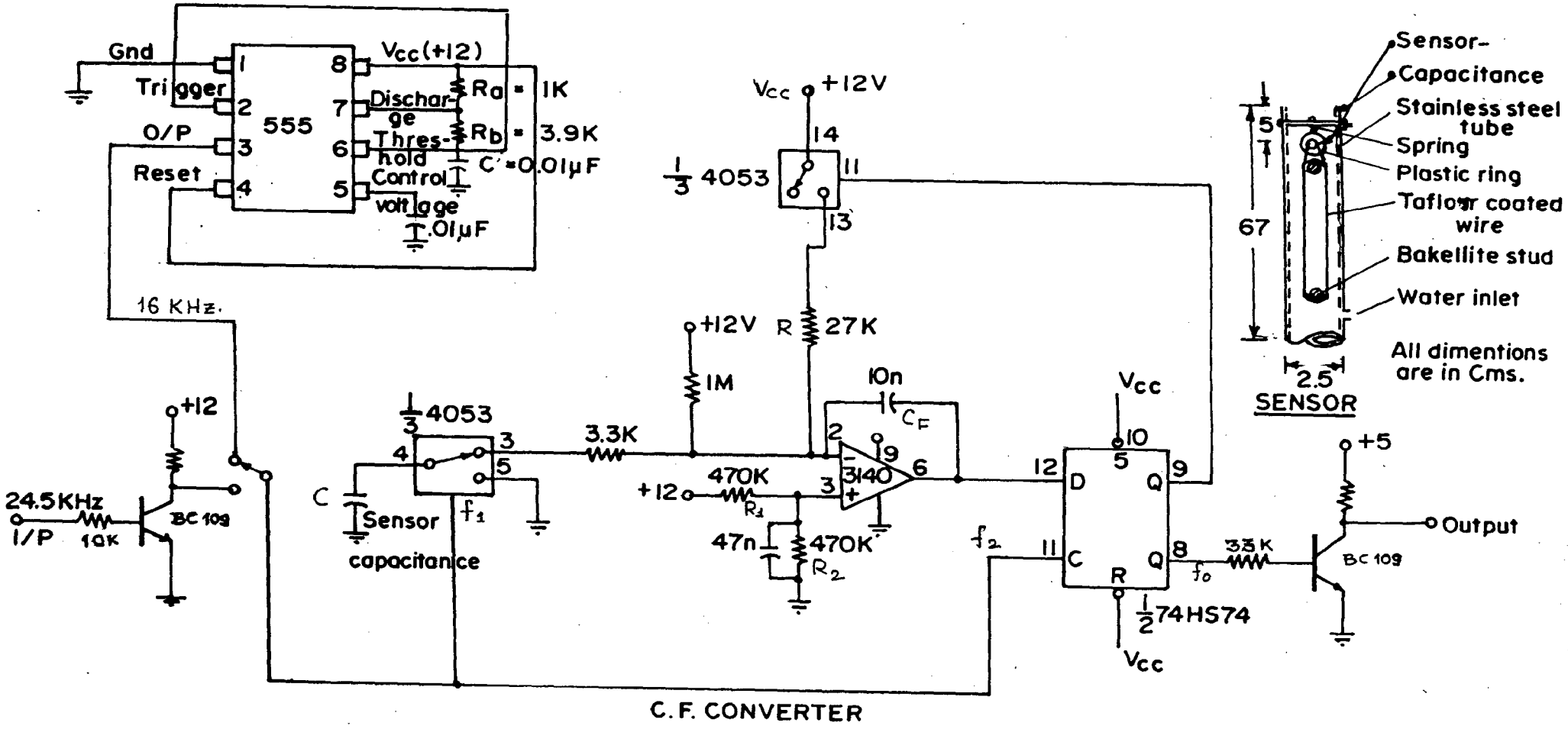


FIG. 3.2-1 WATER SENSOR AND C.F. CONVERTER.

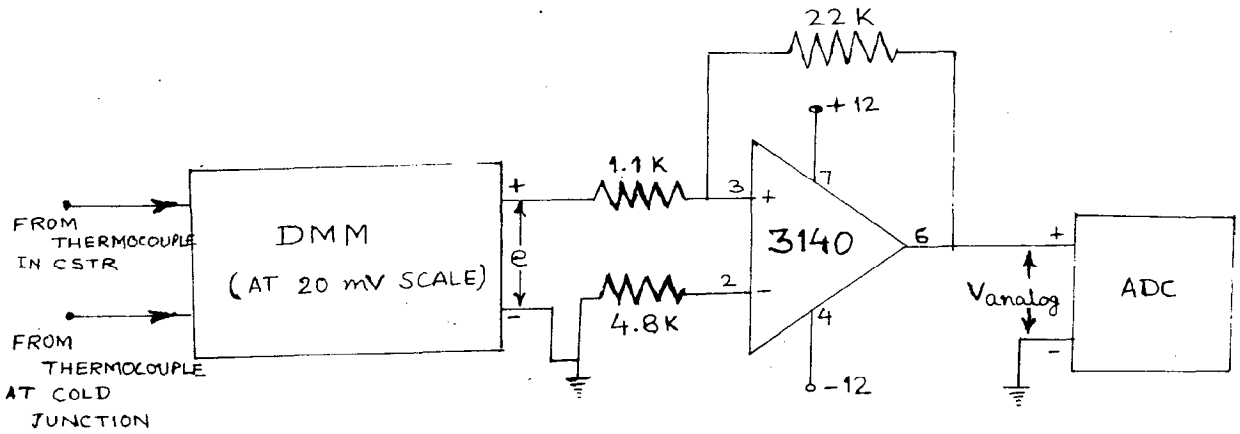


FIG. 3.1.1 : SIGNAL CONDITIONING FOR THERMOCOUPLE

For practical purposes, e.m.f (μv) = 40 x temp (^{o}C).

The temperature of water in the CSTR can reach to a max. of 100^{o}C (as the CSTR is open to atmosphere), at which the thermocouple O/P would be 4.25 mv. This voltage is amplified by a factor of approx. 1000 to make it suitable for ADC. The amplification stage is shown in Figure 3.1.1. The DMM, (model 177), provided by Keithley, Ohio (USA), is kept at 20 mv scale. It gives an O/P proportional to the input voltage (i.e. thermocouple O/P), e (mv) as

$$e'(\text{volt}) = \frac{e(\text{mv})}{20}$$

The OP-AMP 3140 amplifies this voltage by a factor of 21 and feeds it to ADC. Thus the I/P to ADC is given by

$$V_{analog} = \frac{e^{(mv)}}{20} \times 21 \text{ volts}$$

$$= 1.05\ e(\text{mv}) \text{ volts}$$

Thus, V_{analog} varies with water temp. in the range 0V to 4.46V, which can be read by up through ADC to monitor temperature of the water.

3.2 LEVEL TRANSDUCER AND SIGNAL CONDUTIONING [11]

A capacitive water-level sensor using taflon coated wire and a capacitance to frequency converter is developed and is described here. The sensor converts water level to electrical capacitance, hence the method of measuring the capacitance

must be unaffected by conductivity. In addition, a digital signal is required for interfacing to the μp. These requirements are met by developing a capacitance to frequency (C-F) converter.

3.2.1 SENSOR

The sensor, as shown in Fig. 3.2.1, consists of a taflon coated wire mounted inside a stainless steel tube. The wire is wrapped around a bakellite stud at the bottom of the tube and the ends are joined above the maximum water level, thereby doubling the capacitance and avoiding the problem of sealing the end of the wire.

A bakellite stud is used to space the wires at the top of the tube where they are anchored to a rigit plastic ring and tensioned by an spring.

Water forms one plate of this capacitor and the wire the other, while the taflon insulation is the dielectric. The plate area and therefore the capacitance change linearly with water level.

3.2.2 C-F CONVERTER

Fig. 3.2.1 shows a C-F converter along with the sensor. An integrator balances charge on its feedback capacitance C_F by using a clocked D flip-flop and switch to dump a precise amount of charge into it whenever its O/P voltage rises above

the flip-flops logic level. The clock to the D flip-flop can be given by either a 555 timer used in astable mode (16 KHz) or through the A/D converter module (24.5 KHz). The either option can be selected through a jumper.

A signal current is provided by repeatedly charging the sensor capacitor C from the integrator input and then discharging it.

Equating the signal current f_1 Vcc/2 to the feedback current f_o Vcc /2Rf_2, the O/P frequency f_o is given by

$$f_o = f_1 f_2 RC,$$

where f_1 is the frequency at which the capacitor is switched, f_2 the D-flip-flop clock frequency and R includes the resistance of the switch.

For $f_1 = f_2 = f_c$, say

$$f_o = f_c^2 RC$$

The max. possible O/P frequency is $f_2/2$.

Hence, $2f_1RC < 1$.

The feedback capacitance C_F should be large enough to prevent the OP-AMP O/P from saturating. Allowing a max. O/P voltage change of $V_{cc}/4$ when charge is dumped into the input, it is required that

$$C_F > 2/f_2R.$$

However, C_F must not be so large that the O/P voltage change can ever fail to be detected by the flip flop.

In the circuit shown in Fig. 3.2.1, the clock to the flip-flop can be given through two options -

1) A 16KHz clock generated by I.C.555 timer, used in astable mode.

2) A 24.5 KHz clock from the A/D converter module, discussed in section 4.2, amplified to 12V. Either option can be selected through a jumper.

The C-F converter O/P is a train of +12V pulses, which is brought to TTL level by the transistor BC 109.

CHAPTER - IV

HARDWARE DEVELOPMENTS

4.1 SCR FIRING CIRCUIT

DESCRIPTION :

This is a control element for controlling energy input to heater and then to control heat input to the system through the bridge as dimended by the controller.

4.1.1 OPERATION OF THE THYRISTORS

Fig. 4.1.1 shows the complete firing circuit along with the single phase half controlledrectifier bridge. T_1 and T_2 are the two thyristors(OE 2506) and D_1 and D_2 are the two diodes (OE 2506), each with a maximum current rating of 25A and peak inverse voltage of 600 V. The firing circuit is described in the next section.

A sinusoidal input (220V, 50 Hz) is applied between terminals A and B. When A is positive with respect to B, diode D_1 is forward biased. Now if T_1 is fired at an angle α (0 to 180°), a current flows through T_1- load - D_2. Similarly when A is negative with respect to B (i.e. in the negative half cycle of the sinusoidal input), a current will flow through the path T_2- load- D_1, when the thyristor T_2 is fired. The current through the load (auxiliary heater in this case) is always unidirectional.

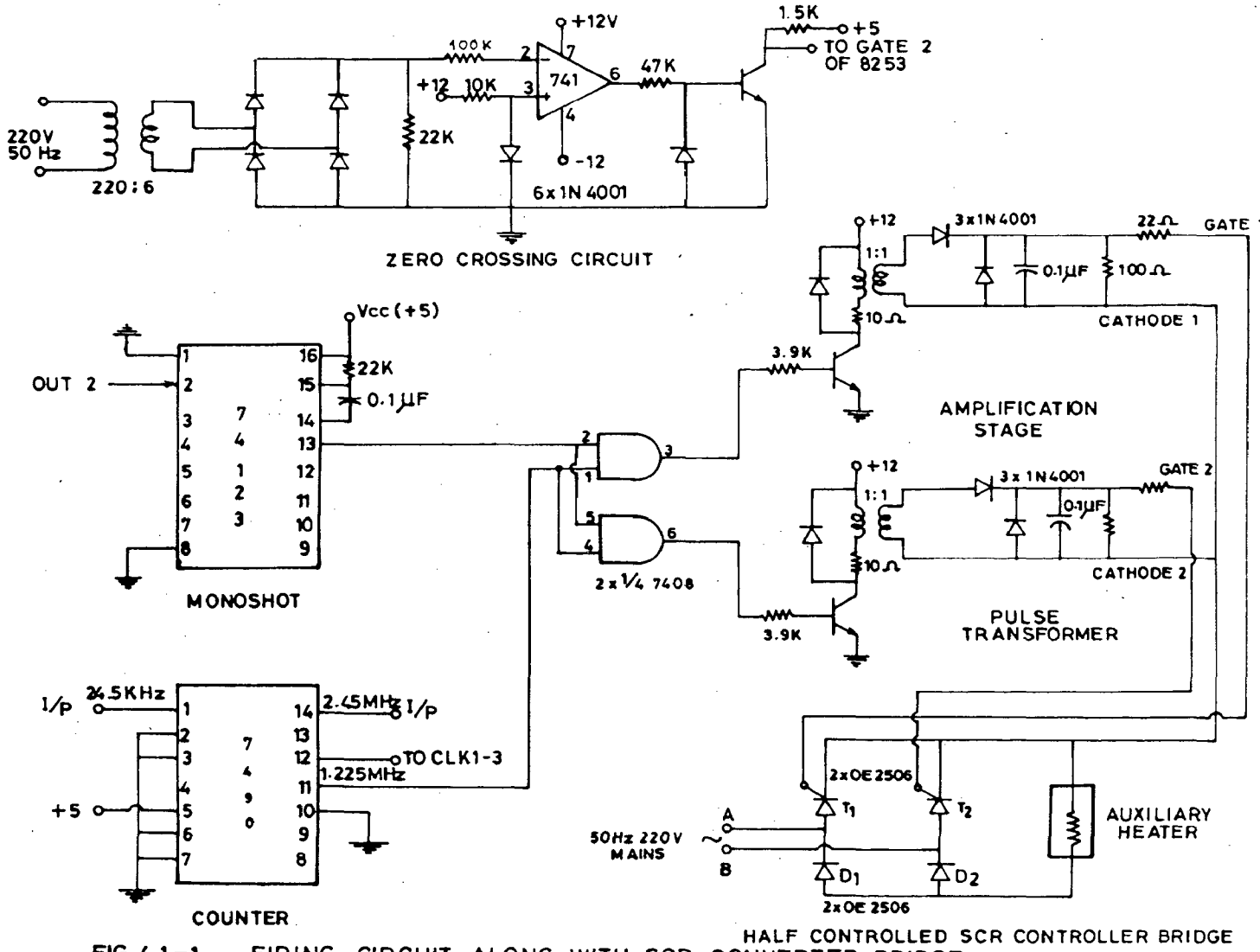


FIG. 4.1-1 FIRING CIRCUIT ALONG WITH SCR CONVERTER BRIDGE

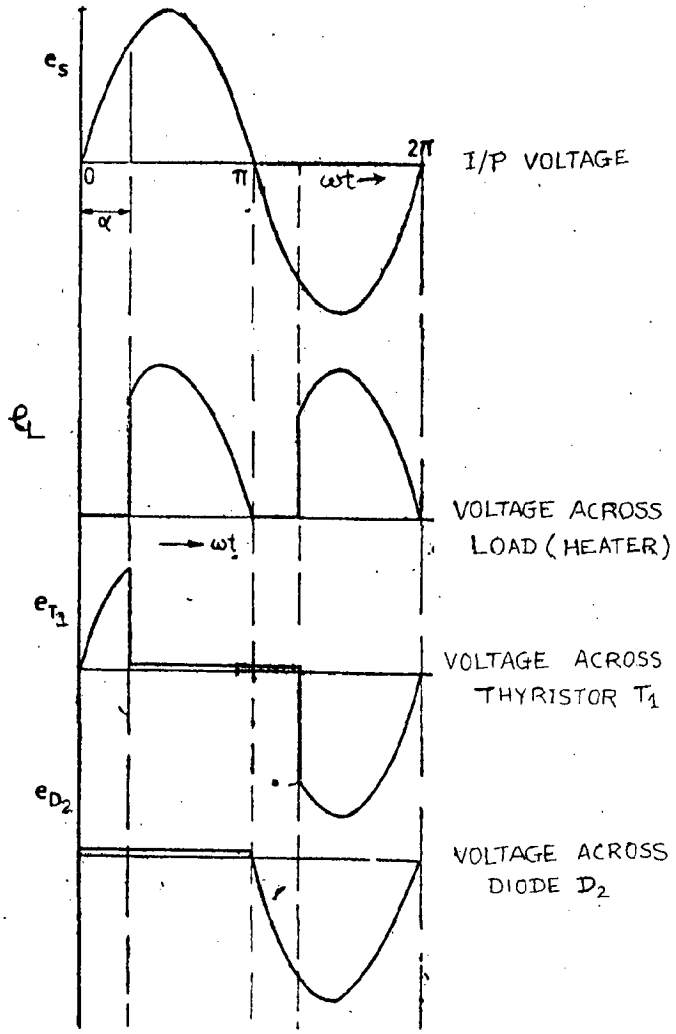


FIG 4·1·2 : VOLTAGE WAVE FORMS IN SEMICONVERTER BRIDGE.

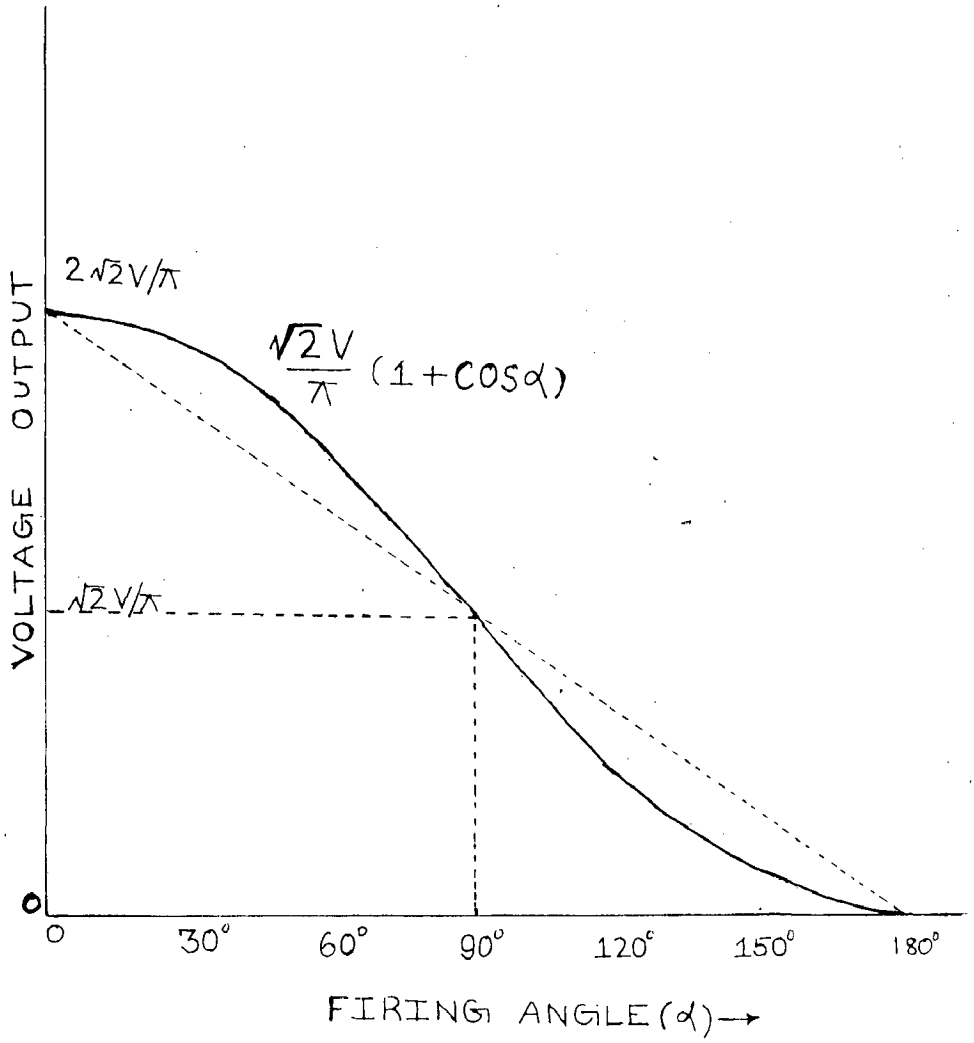


FIG-4.1.3 : OUTPUT VOLTAGE OF SEMICONVERTER AT DIFFERENT FIRING ANGLES

The variations of supply voltage, voltage across heater, voltage across thyristors, voltage across diodes and the current through heater with respect to firing angle are shown in fig. 4.1.2. The heater is assumed to be purely resistive.

The expression for average voltage across the load may be derived as -

$$V_{av} = \frac{1}{\pi} \int_{\alpha}^{\pi} V_m \sin \omega t \, d(\omega t)$$

$$= \frac{1}{\pi} V_m \left[-\cos(\omega t) \right]_{\alpha}^{\pi} = \frac{V_m}{\pi} (1 + \cos \alpha)$$

The variation of V_{av} with the firing angle α is shown in fig. 4.1.3.

4.1.2 DESIGN OF FIRING CIRCUIT

ZERO CROSSING DETECTER

A zero crossing detecter as shown in fig. 4.1.1 is required to indicate the zero crossing (0° or 180°) of the input sine wave because the firing angle α is to be measured from these instants.

The input sine wave (220V) is stepped down through a transformer to 6V, which is then rectified by a diode bridge and applied at the inverting input of the comparator, the non-inverting input of which is at 0.7 V (voltage drop across diode). When the input at inverting input exceeds this voltage, the O/P of comparator gives - V_{cc}(i.e. -12V). The

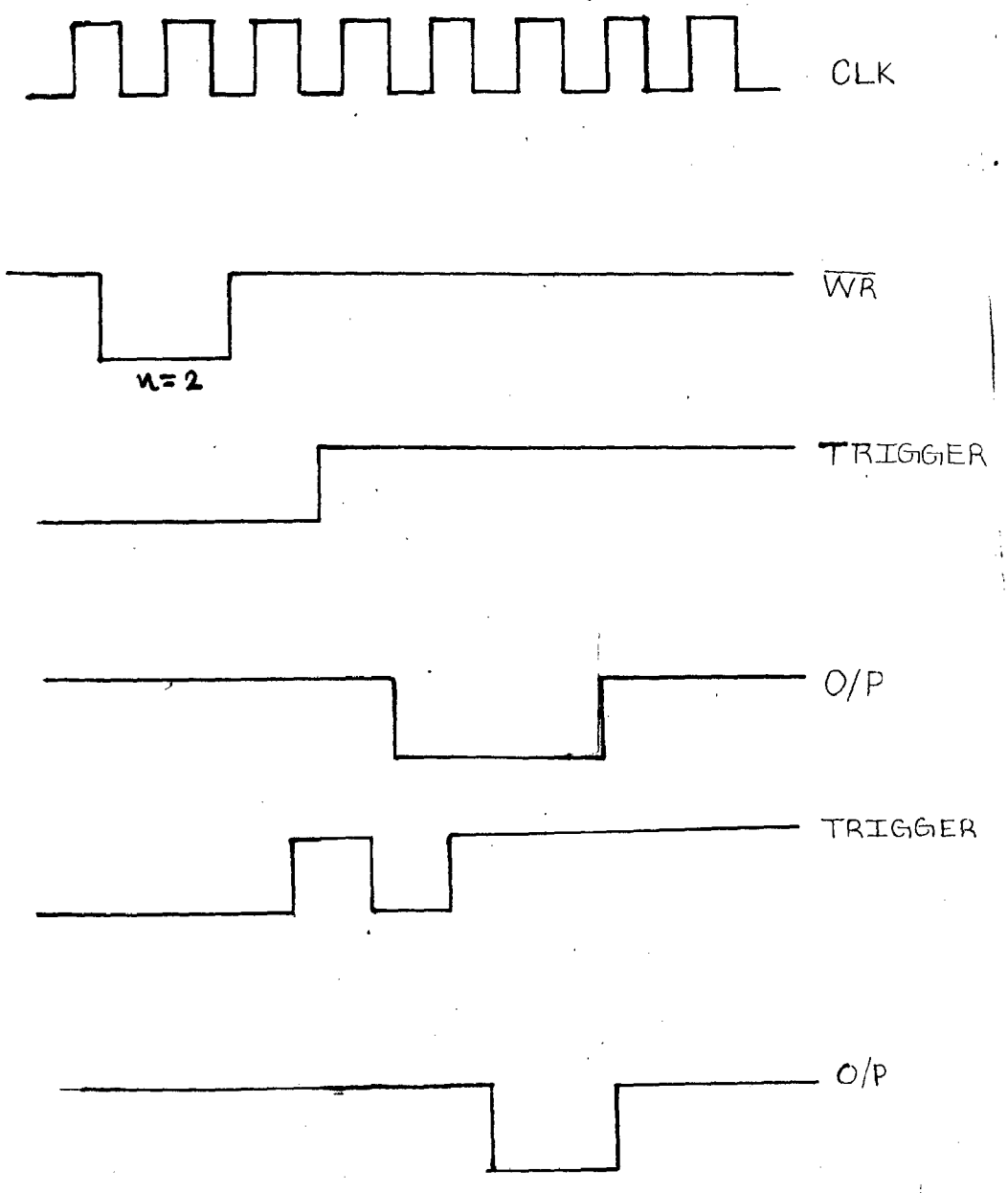


FIG - 4.1.4 WAVE-FORMS FOR TIMER IN MODE-1

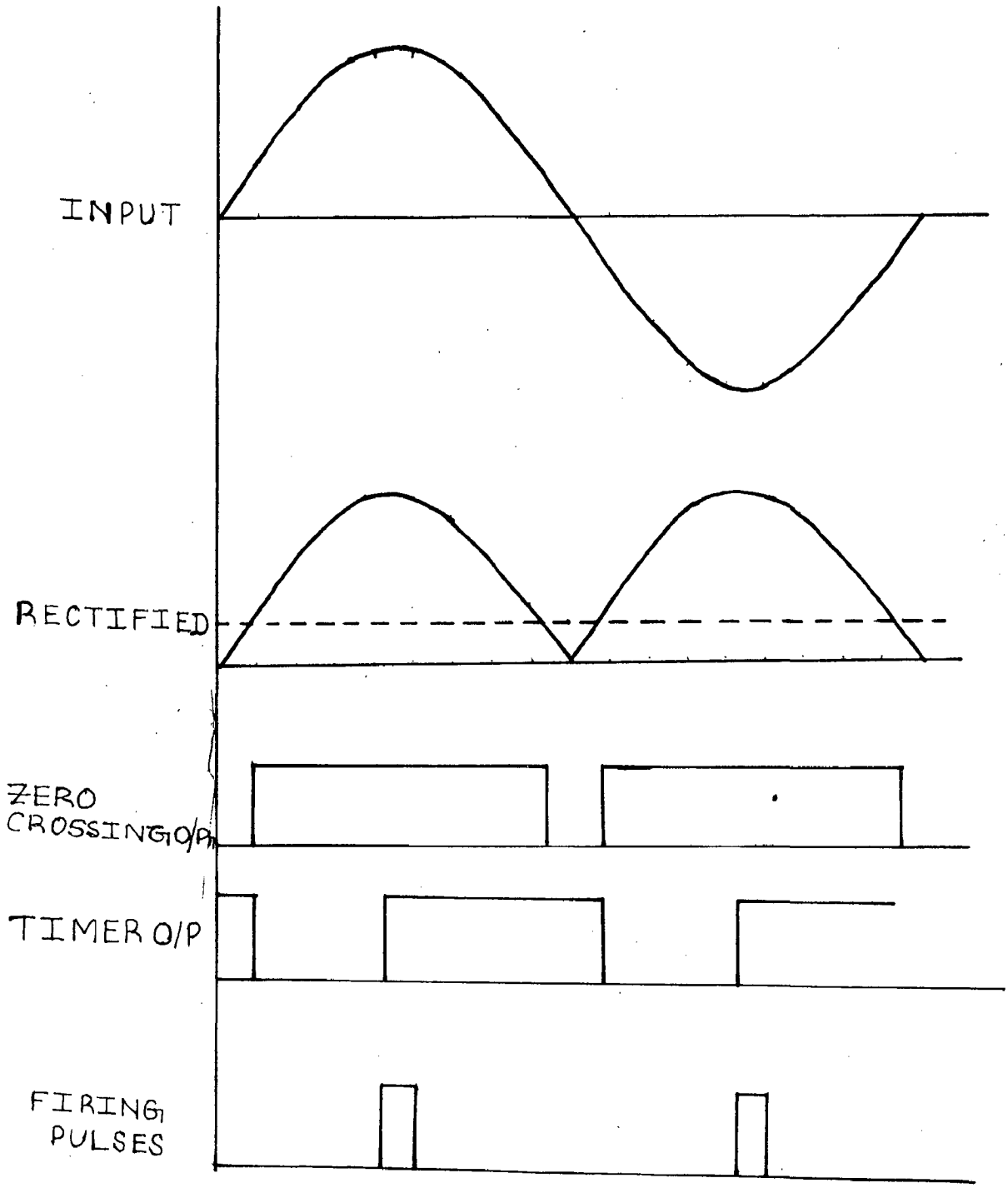


FIG. 4.1.5 : WAVEFORMS AT VARIOUS POINTS IN SCR FIRING CIRCUIT

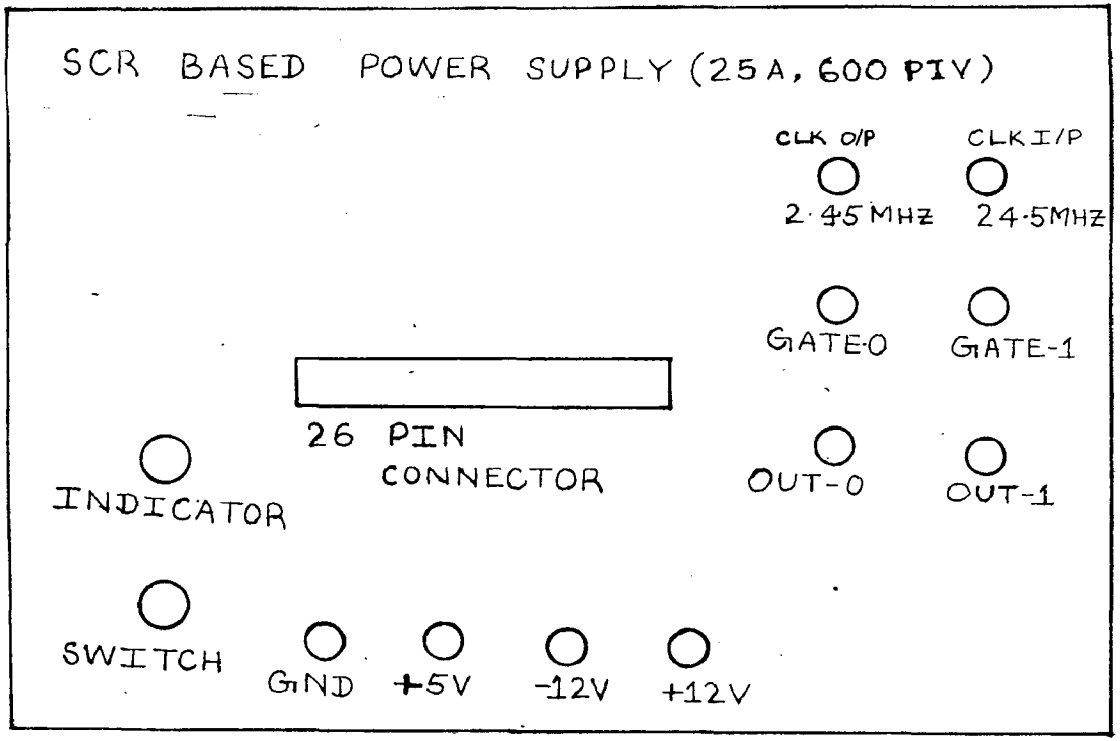


FIG 4·1·6 FRONT PANEL OF SCR MODULE

diode at the base of the transistor limits this voltage to -0.7V by conducting in the forward biased direction. Thus, the transistor goes to cut-off region, thereby giving an O/P of 5V at its collector.

When the voltage at inverting input is below that t at the non-inverting input of the comparator, the O/P of comparator goes to $+V_{cc}$ (i.e. +12 V) which brings the transistor into saturation. Therefore an O/P of 0.3 V(logic 0) at collector of the transistor is obtained. This O/P can be neglected and is taken as OV. Thus, at each zero crossing of the input sine wave, (ie. 0^{o} or 180^{o}) a low to high transition (TTL level) at the collector of the transistor is obtained.

The output of zero crossing detecter is shown in Fig. 4.1.5. This output is given to GATE 2 of 8253, with counter 2 initialized in mode 1(i.e., programmable one-slot) and already loaded with a count equivalent to the firing angle. The clock to all the counters of 8253 is given by dividing PCLK (2.45 MHz) by 2 using the counter 7490. Thus, the CLKO, CLK 1 and CLK 2 of 8253 are given 1.225 MHz clock. The output (OUT 2) waveform in mode 1 is shown in figure 4.1.4. After a delay (N/1.225 us; N being the count loaded into COUNTER 2) equivalent to the firing angle α from the 0^{o} or 180^{o} reference, it gives a low to high transition and then a high level till the next zero crossing, at which retriggering

of COUNTER 2 takes place. This output is not strong enough to turn on an SCR. So an amplification stage, described in next section is required.

AMPLIFICATION STAGE

The rising edge obtained at OUT 2 of 8253 triggers a monoshot (74123). The duration of the monoshot's output pulse is governed by the R_{ext} and C_{ext} connected to it and is given by

$$t_w = 0.28\ R_{ext}\ C_{ext}\ (1.0 + \frac{0.7}{R_{ext}})\ ns$$

where R_{ext} is in K ohms and C_{ext} in pF.

With R_{ext} = 22 K ohm.

and C_{ext} = 0.1 uF = 10^5 pF,

t_w = 0.6 ms.

This duration is sufficient to trigger the SCRs. This pulse is AND_{ed} with high frequency pulses of 5 KHz, amplified and fed to the 1:1 pulse transformer. The high frequency AND_{ing} is required to avoid the saturation of the pulse transformer and is obtained by dividing the 24.5 KHz. input to the module by 5 through the divide - by-5 counter of 7490. The 24.5 KHz. input is obtained from the A/D scanner card, described in section 4.2. The pulse transformer provides physical isolation between the contral circuit and the power circuit. The modulated pulses reduce the gate dissipation in the SCR. A diode is connected across the pulse transformer's primary to avoid the saturation of pulse transformer.

Gate protection is required for over-voltage and over-current. Because of the low power level of the control circuitary, this protection can be provided by simple means. The gate can be protected against overvoltage by connecting a resistance in series with the pulse transformer input. A common problem encountered in the SCR circuitary is spurious firing of the device. Trigger pulses may be induced at the gates due to turn on or turn off of a neighbouring SCR or transients in the power circuit. These undesirable trigger pulses may turn on the device, thus causing improper operation of the circuit. Gates are protected against such spurious firing by using shielded cables or twisted gate lead connections. A capacitor is connected across the gate to cathode to bypass the noise pulses.

The waveforms at various points are shown in fig. 4.1.5 and the front panel of the module developed is shown in fig. 4.1.6.

4.2 24-CHANNEL A/D SCANNER CARD

DESCRIPTION :

This module has been developed to convert the analog voltage, obtained by thermocouple after signal conditioning, into digital equivalent. One channel is required for the present work. The other 23 channels may be used for further developments. Moreover, it provides four clock outputs of

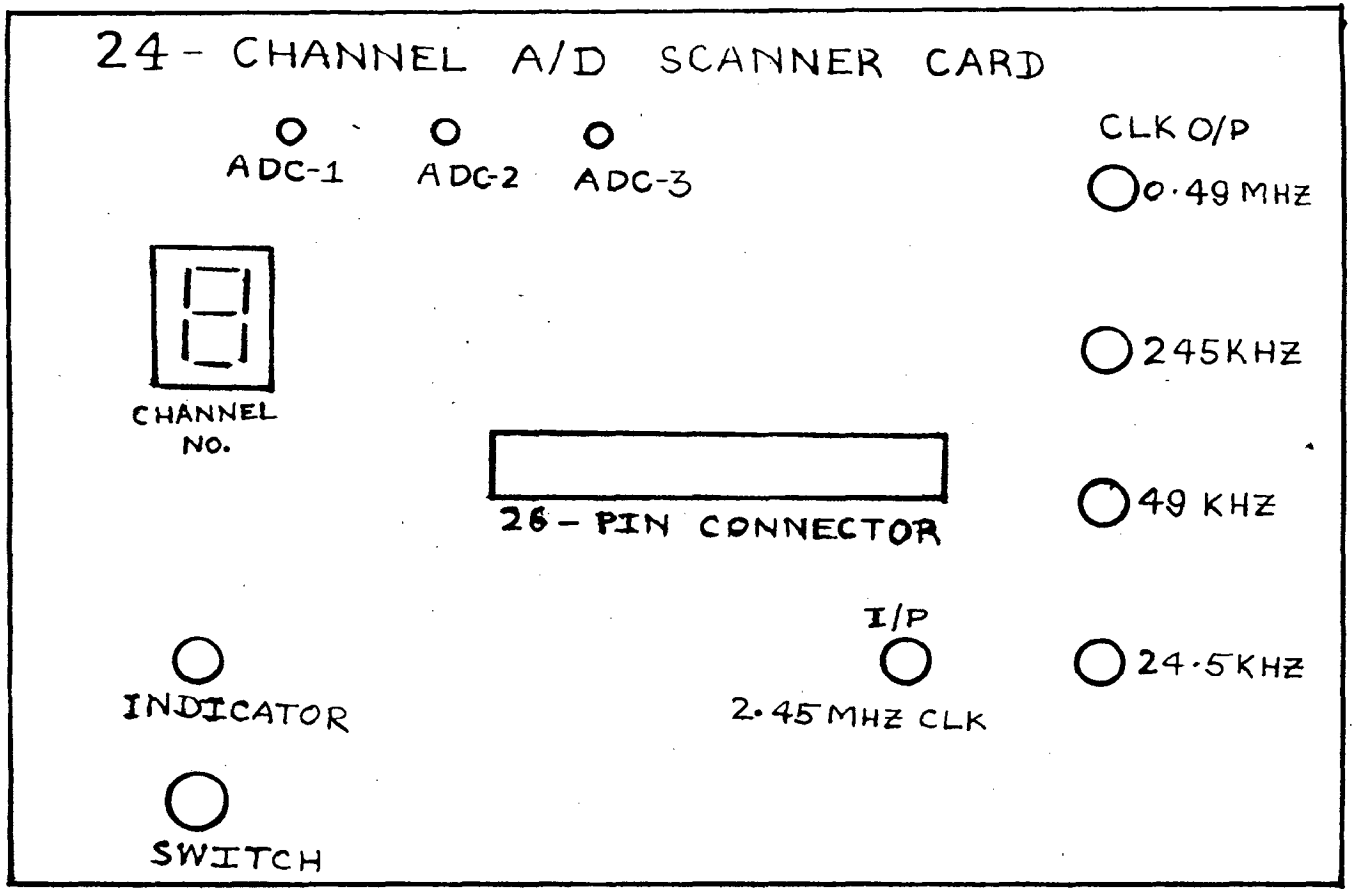


FIG - 4·2·1(a) FRONT PANEL OF ADC MODULE

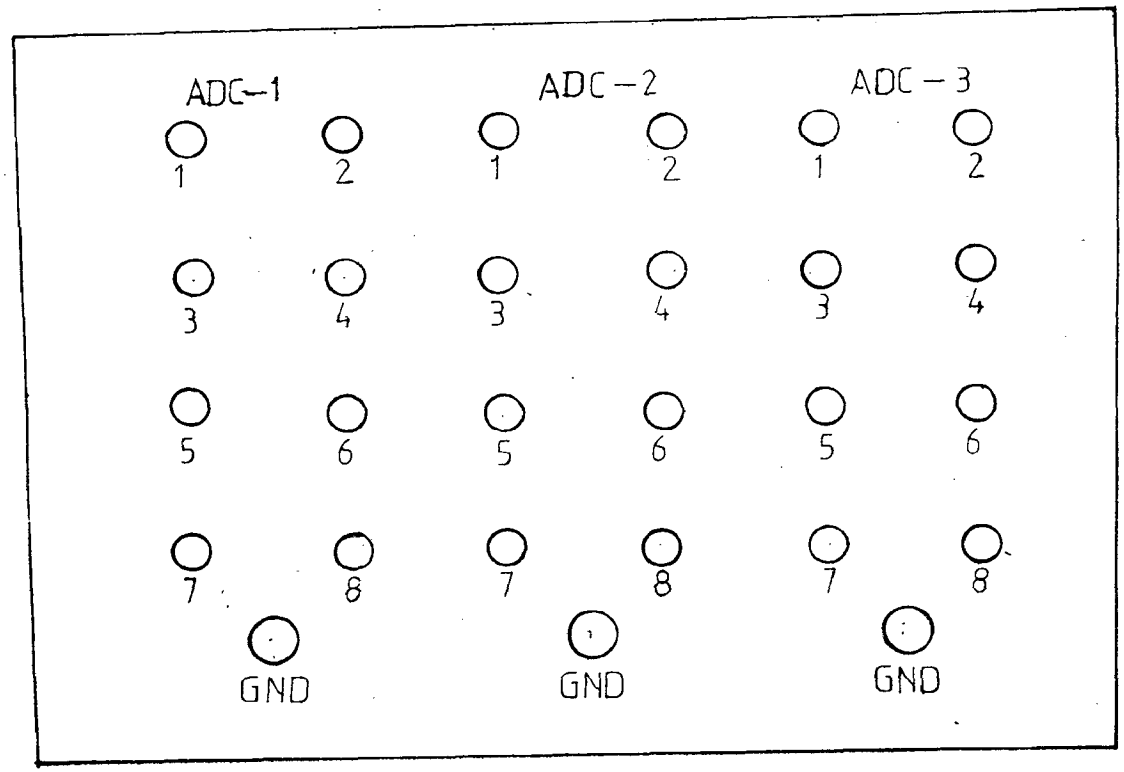


FIG. 4.2.1 (b) – BACK PANEL OF ADC MODULE

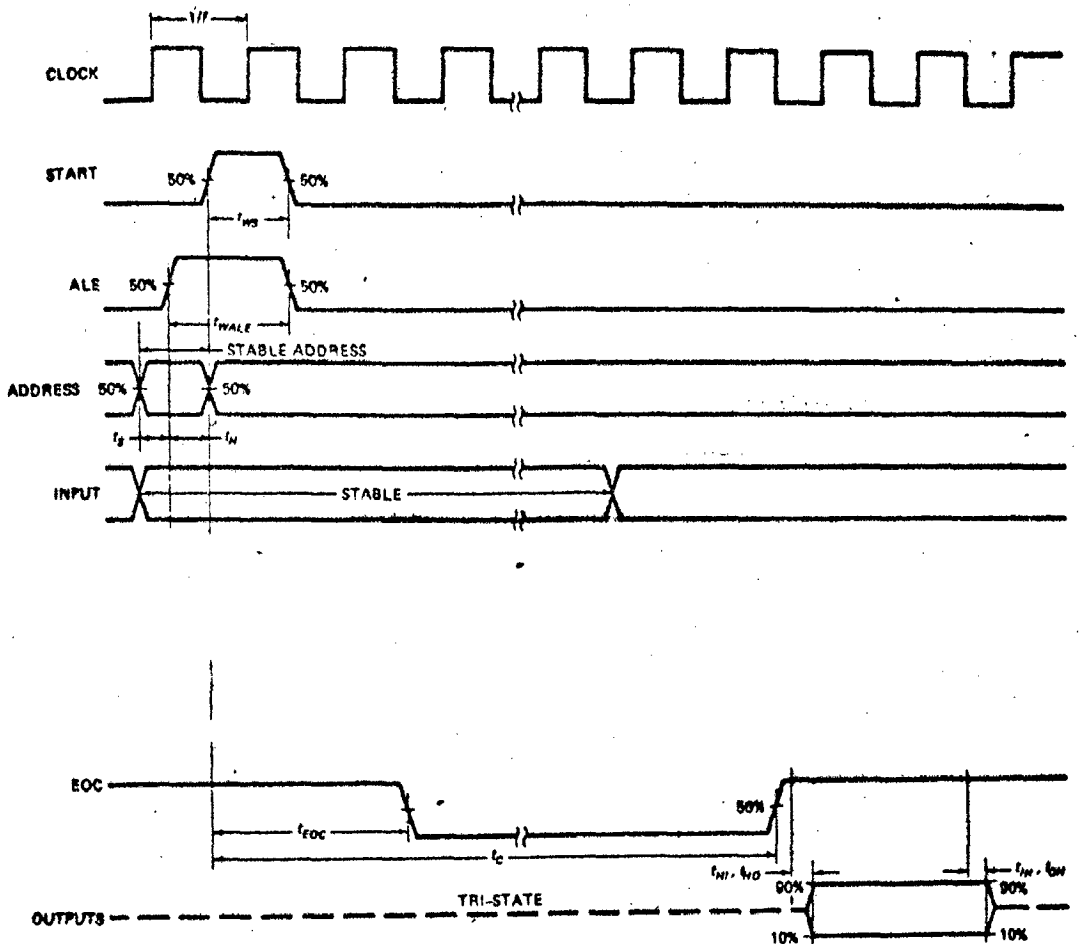


FIGURE 4.2.2 Timing waveforms for the ADC0809

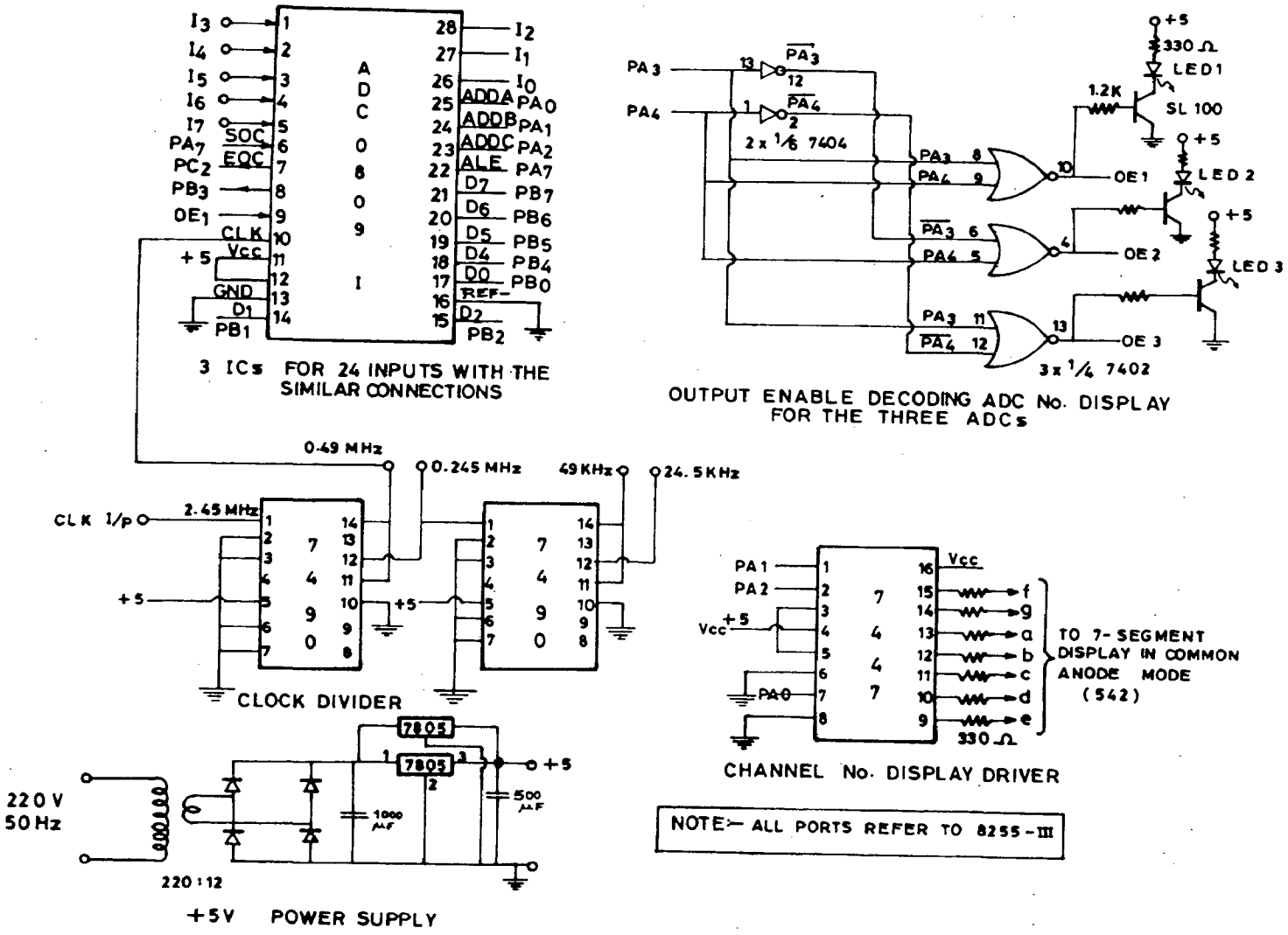


FIG. 4.2 - 3(a) CIRCUIT DIAGRAM FOR A/D SCANNER CARD

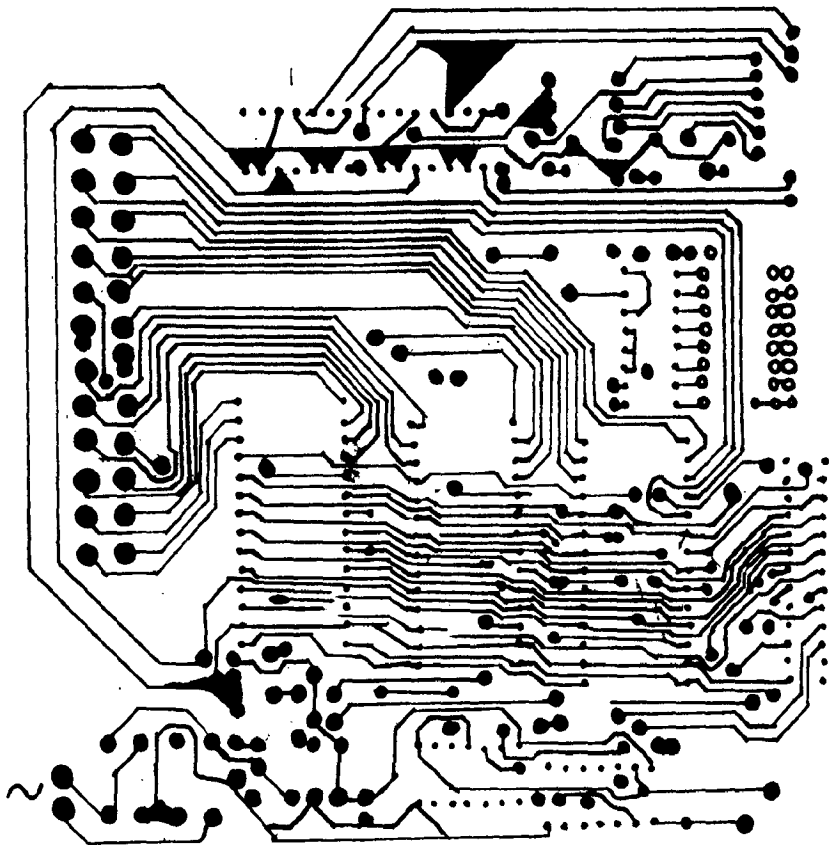


FIG. 4. 2. 3 (b) – ADC CARD PCB LAYOUT

different frequencies - 0.49 MHz, 245 KHz, 49 KHz and 24.5 KHz, the only input being 2.45 MHz clock. An in-built 5V D.C.power supply is also included. The front panel of this module is shown in Fig. 4.2.1(a). The 24 inputs can be provided from the back panel, which is shown in fig. 4.2.1(b). Three 8-channel, 8 bit A/D converter ICs (ADC 0809) are used to meet the above requirement. The channel no. and the corresponding ADC in operation are indicated by the seven segment display (0-7) and the LEDs respectively on the front panel.

4.2.1 WORKING OF ADC 0809

The pin deg. of the A/D converter,ADC 0809 and the pin descriptions are given in Appendix C. As can be seen from the timing dig. shown in fig, 4.2.2, an Start of Conversion and Address Latch Enable pulse is to be sent after placing the address on the address lines to initialize the ADC. The End of conversion, which is normally high goes low after same delay. It remains low during the conversion period, which is about 100 μs. When it goes high indicating the conversion is over, the digital equivalent can be inputted.

4.2.2 CIRCUIT DESCRIPTION

The circuit dig. of the A/D converter scanner card is shown in fig. 4.2.3(a) and layout of its PCB is shown in fig. 4.2.3(b). The whole circuit dig is divided into various parts as follows:

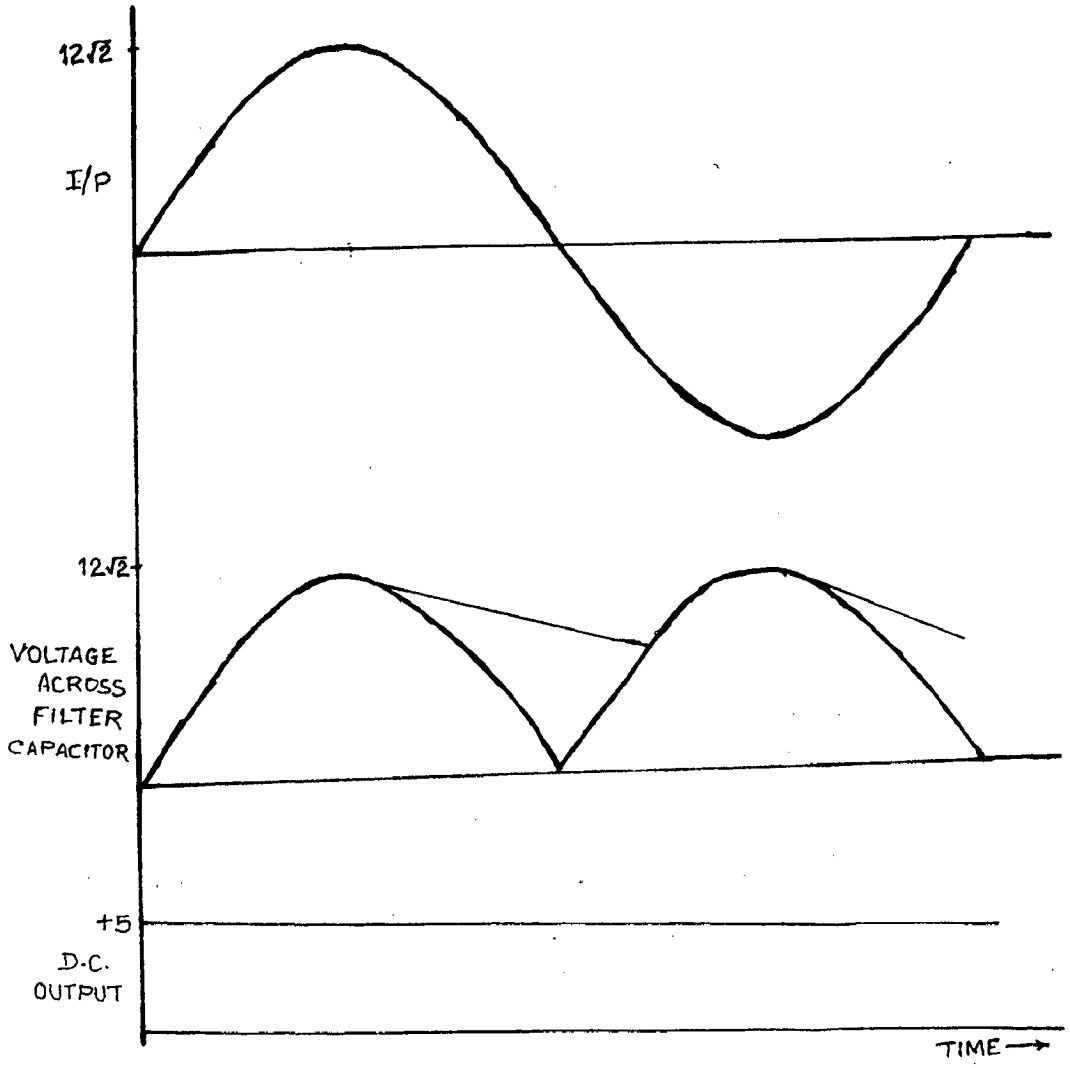


FIG. 4.2.4 WAVEFORMS FOR +5 V D.C. POWER SUPPLY

(a) +5V DC SUPPLY:

A 12V sinusoidal input is rectified by the diode bridge. The capacitor at the O/P of this bridge is charged to the peak voltage of the rectified wave, when this reaches to its peak value. The capacitor O/P decays slowly till the next peak of the rectified wave appears. Thus an O/P as shown in fig. 4.2.4 across the capacitor, i.e. at the I/P of voltage regulator 7805, is obtained. The 7805 then gives a constant +5 V DC supply. Two 7805 voltage regulators are used in parallel so as to increase the o/p current rating to 1A (each 7805 having 500 mA o/p current rating). This is used to supply +5V to all the components on the card.

(b) CLOCK DIVIDER :

A clock I/P of 2.45 MHz is divided by five, ten, fifty and hundred to give 0.49 MHz, 245 KHz, 49 KHz. and 24.5 KHz. by using the two 7490 counters. Each 7490 has one divide-by-five counter and a divide-by-two counter. The 2.45 MHz I/P is divided by 5 to give 0.49 MHz by using one 7490. This o/p is fed to the divide-by-two counter of the same 7490 so as to give 245 KHz. Similarly, other 7490 gives, with 245 KHz as an I/p, the two frequencies

of 49 KHz and 24.5 KHz. These clock points are brought to the terminals on front panel, shown in fig. 4.2.1(a). The 0.49 MHz clock is used for ADC 0809 chips.

(c) ADC 0809 INTERFACING

The I/P is applied on any of the 8 lines I_0-I_7. The I/P to be scanned is addressed by ADDC-ADDA, connected to PA2-PA0 of 8255-III through a 26 pin connector. The address is latched by ALE through PA7. An start of conversion signal is sent through PA6 and the End of conversion is polled through PC2-PC0 for the three ADCs (ADC 1-3). The data is inputted through PB.

7402 and 7404 are used for decoding which ADC's OE is to be made active (high). PA3 and PA4 lines are used for this purpose. The decoding is shown below:

PA4	PA3	ADC No.SELECTED
0	0	ADC-1
0	1	ADC-2
1	0	ADC-3
1	1	NONE

So, the 8-I/P channels of ADC-1 are addressed by 00-07 (for I0-I7), for ADC-2 (08-0F) and that for ADC-3 by 10-17, through PA.

The 0.49 MHz CLK to the three ADCs are provided by the o/p of one of the 7490s, as discussed earlier. The reference voltage is kept +5V and the -ve reference voltage is grounded.

(d) DISPLAY

The 3 LEDs on the front panel indicate which ADC is currently working. The OE of the three ADCs are given to the base of 3 transistors (BC-109) seperately. The OE of the ADC under operation will go high and drive the corresponding transistor. The collector current flowing through the ON transistor will glow the corresponding LED.

A 7-segment display in common anode-mode (592) displays the channel no. of the selected ADC in operation. Since the channel no. is addressed by PA2-PAo, so the same lines are used for displaying the channel no. (0-7). Thus, the three I/Ps of 7-segment display driver (7447) are connected to PA2-PAo (PAo being least significant) and the fourth I/p (most significant) is grounded.

Thus, the 3- LEDs and the seven segment display indicate which of the 24-channels is working.
Output of the ADC is:

$$\text{Digital Equivalent} = \frac{V_{analog}}{255} \times V_{ref}.$$

4.3 STEPPER MOTOR DRIVING CIRCUIT

DESCRIPTION :

The stepper motor is used to control the level of the water in the tank by operating a value which controls the outlet of the water from the tank.

Stepper motor is one of the most suitable devices to convert digital pulses into precise incremental rotary motion, and can be effeciently used to control the valve and thus flow with the help of a up issuing control pulses.

The specification of the stepper motor used for the above experiment are:

i) Type: Permanent Magnet D.C., Bifiler wound SM

ii) Step angle: 1.8 ± 5 percent non-cumulative,

iii) Steps/Revolution : 200,

iv) Speed: 0-10,000 steps/sec, i.e. 0-3,000 RPM.

4.3.1 WORKING OF STEPPER MOTOR

The PM stepper motor is shown in Fig. 4.3.1. It consists of two stator windings, viz. A and B and a rotor having the magnetic poles N and S. When any one of the stator winding is energised, the corresponding magnetic poles are generated in the stator. The rotor (permanent magnet) hence positions itself such that its poles lock with the corresponding stator poles. When the two windings are energised simultaneously, the rotor positions itself along the direction of the resultant magnetic field.

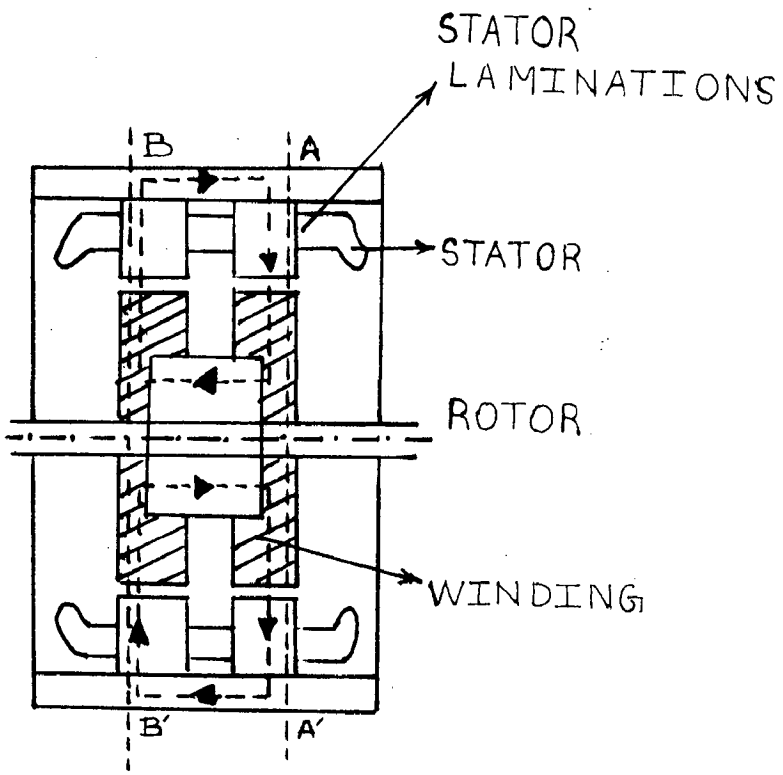


FIG. 4.3.1 : PM STEPPER MOTOR

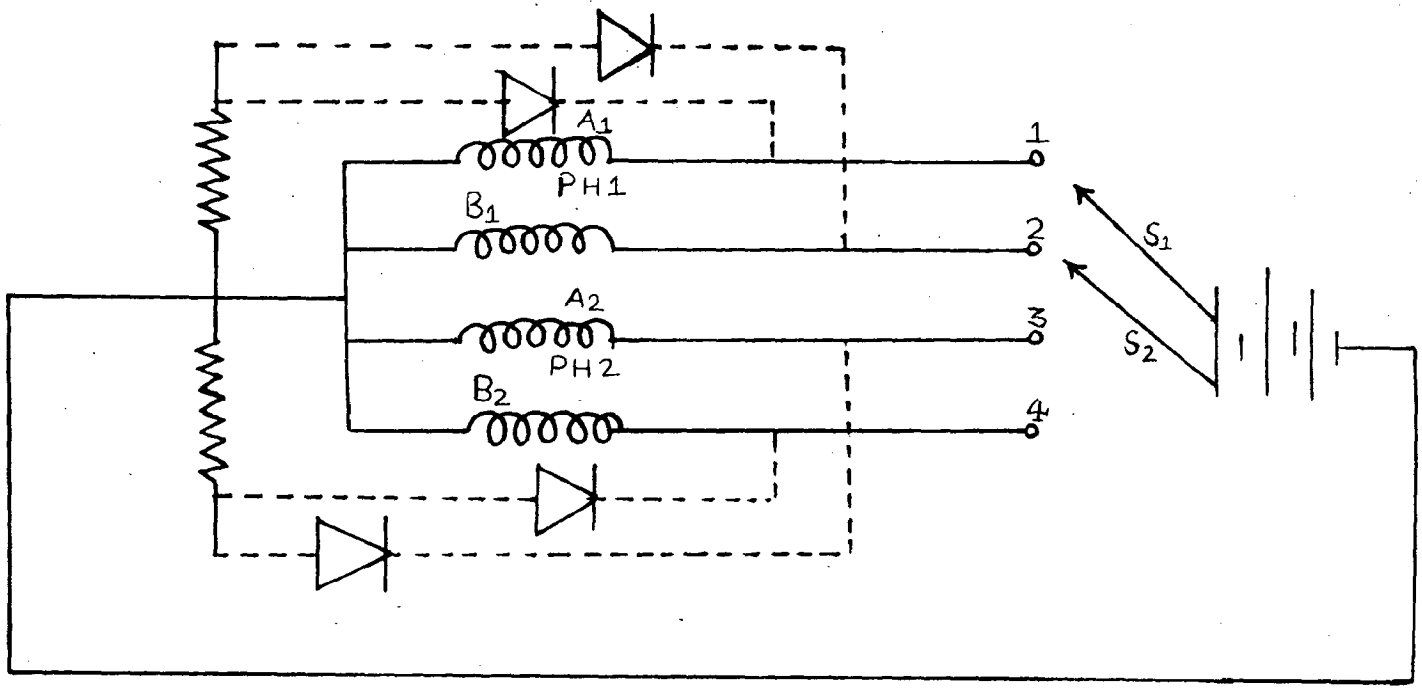


FIG. 4·3·2 SWITCHING DIAGRAM FOR EXCITATION OF WINDINGS

The different combinations of excitation of stator windings along with the corresponding rotor positions are shown in Fig. 4.3.2.

The motor used has 1.8° step angle, it has 50 teeth on the rotor and 8 main poles on the stator.

The step angle, $\theta_s = \frac{360}{N_r K_s}$

where N_r = No. of rotor teeths,

K_s = Excitation sequence factor.

The following 3 modes of operation of PM stepper motor are possible:

i) Single phase mode :

In this mode, only one of the motor windings is excited at a time. There are 4 steps in excitation sequence.

ii) Two phase mode:

Both the phases are excited at a time. This mode also consists of 4 steps in excitation sequence.

In both these modes, the step angle is 90° and excitation sequence factor is 2. However, the rotor position is 45° away from those in the single phase mode.

iii) Hybrid mode:

It is the combination of both single phase and two phase modes. The voltage +v is applied during certain steps while voltage -V is also applied sometimes. So

this requires a bipolar regulated supply and a pair of SPDT switches. To avoid this, each of the stator windings is splitted into two sections A1, A2 and B1, B2. These sections are wound differentially. These winding sections can now be excited from a unipolar regulated supply. This type of construction of PM stepper motor is called bifilar winding construction. Advantage achieved by doing so is the reduced winding inductance.

Stepping motor being used is PM Bifilar Wound Stepper motor with six leads, operated in Hybrid mode. Each of the two phases has double winding with a centre tap. The advantage achieved by doing so is that switching the supply from one side to another of a phase causes reversal of magnetic polarity without actually reversing the polarity of supply. Four step input sequence gives 1.8°(full) and eight step input sequence gives 0.9° (half) step function.

The switching sequence along with 4 and 8 step input sequence is given in table 4.3.1. This switching sequence will move the shaft in one direction, to reverse the direction of movement, the reverse sequence of the above (i.e. (upwards) is to be followind.

In the present case, however, the 4 step input sequence has been used.

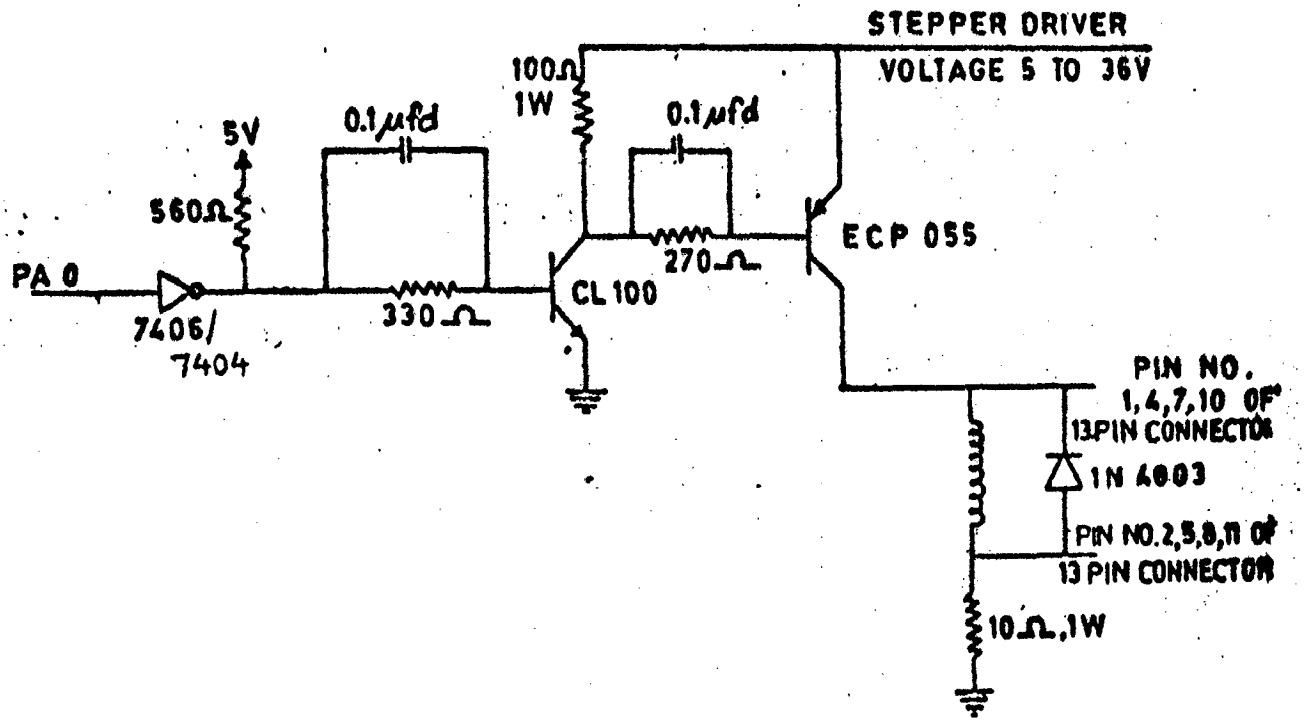


FIG. 4.3.3 – STEPPER MOTOR INTERFACE

Table 4.3.1

SWITCHING SEQUENCE							
4-STEP INPUT				8-STEP INPUT			
PH-1		PH-2		PH-1		PH-2	
A-1	B-1	A-2	B-2	A-1	B-1	A-1	B-2
1	0	1	0	1	0	1	0
0	1	1	0	0	0	1	0
0	1	0	1	0	1	1	0
1	0	0	1	0	1	0	0
				0	1	0	1
				0	0	0	1
				1	0	0	1
				1	0	0	0

4.3.2 NEED OF STABLIZED CURRENT

The torque is directly proportional to the current in winding, which is governed by the d.c. resistance of the winding. As the switching starts, the inductive resistance of the winding, which increases with the frequency of switching, opposes the rise of current to desired level within the time given for one step depending on the frequency of stepping. This is mainly due to L/R time constant of the

winding. The drop in current level causes drop in torque as the speed increases. In order to improve torque at high speeds; it is necessary to maintain current at the desired level. This can be achieved by one of the following methods.

i. By increasing supply voltage and introducing current limiting resistances in each phase. Introduction of resistances improves the time constant of the winding.

ii. By using a constant current source with or without a chopper instead of using a constant voltage source, which will give even better performance.

The stepper motor driver card, provided by Vinytics, is shown in Fig. 4.3.3, with the change that +12 V supply is used for driving stepper motor rather than +5 V.

The pin dig. and the pin descriptions of all the ICs used for developing the hardware modules are given in Appendix - C.

CHAPTER V

DESIGN OF CONTROLLER

A controller is designed to adjust the state of a process as measured by some transducer- the process variable (PV) - to conform to a particular standard value called the set point (SP). The difference between them is termed as the error E, i.e.

$$E = SP - PV$$

A control algorithm for digital computer implementation i.e. a calculation method that produces a control output by operating on an error signal is developed later in this chapter. The different types of control schemes and their control laws (i.e. their behaviour with the error) are also described in this chapter.

5.1 CONTROL SCHEMES

A control system block dig. is shown in Fig. 5.1.1.

5.1.1 PROPORTIONAL CONTROL

The desired mode of control in this scheme consists of manually adjusting the final controlled element corresponding to zero error under average process conditions. The control low for the proportional mode is -

$$P = KE + P_s$$

where $E = SP - PV$,

K = a proportionality constant (gain of the controller),

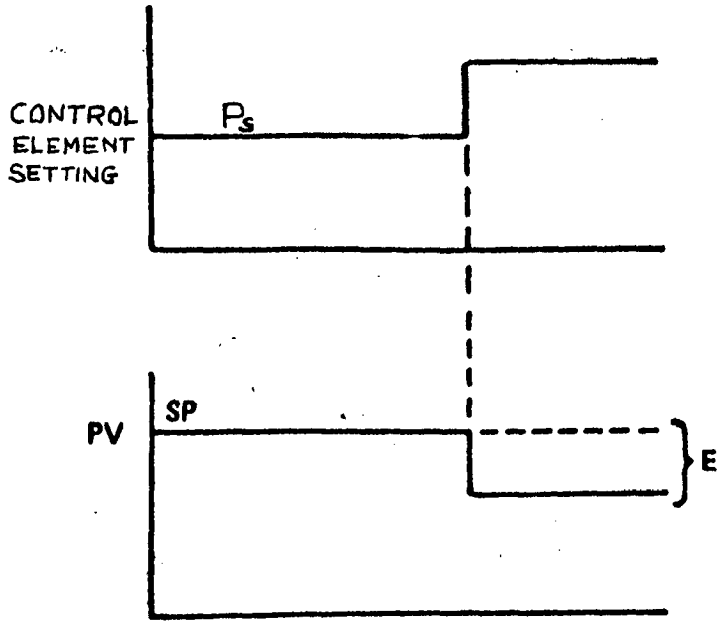


FIGURE 5.1.3 **Response to proportional control.**

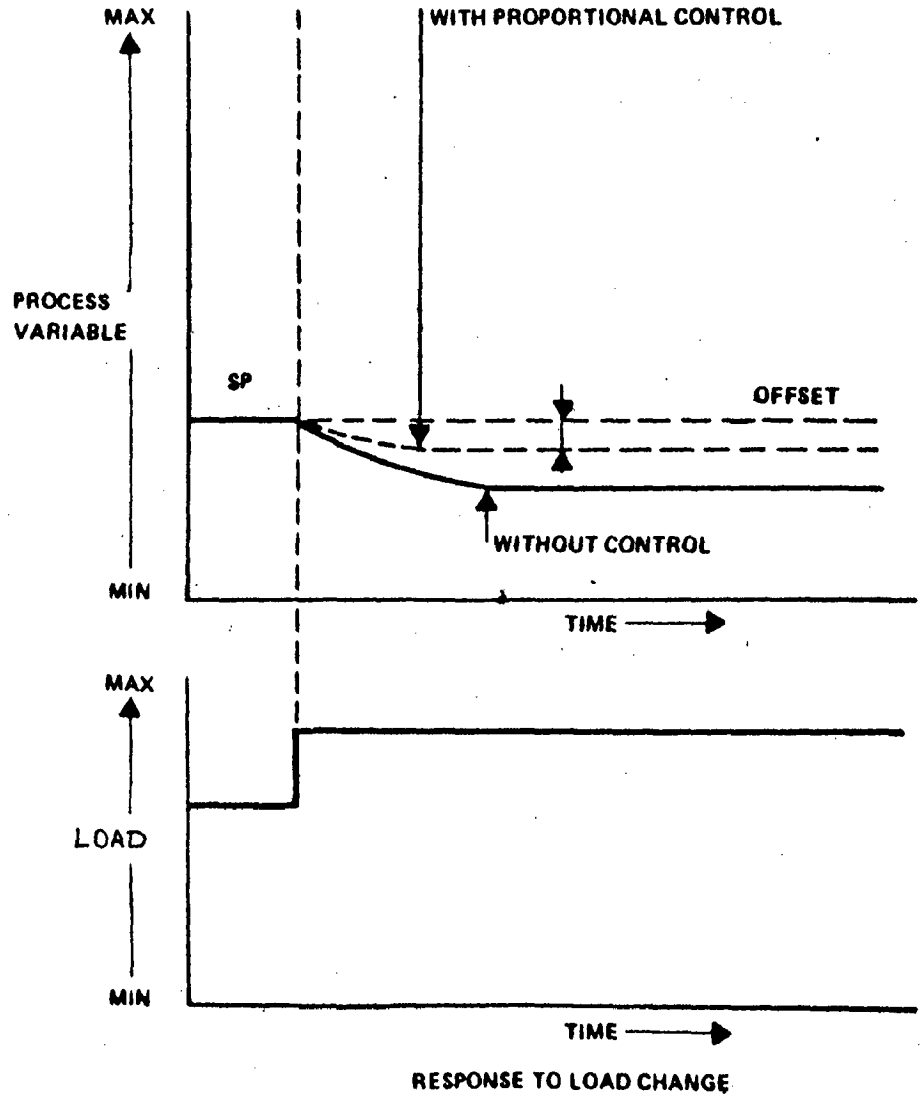


FIGURE 5.1.4 **Load step response.**

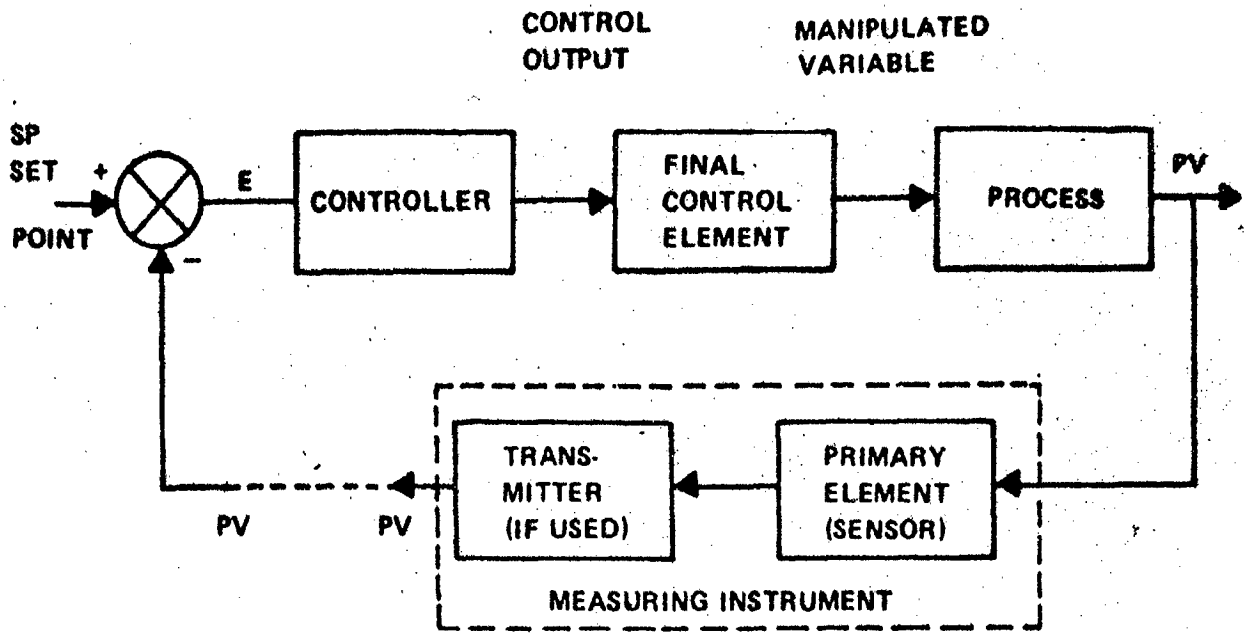


FIGURE 5.1.1 Control system block diagram.

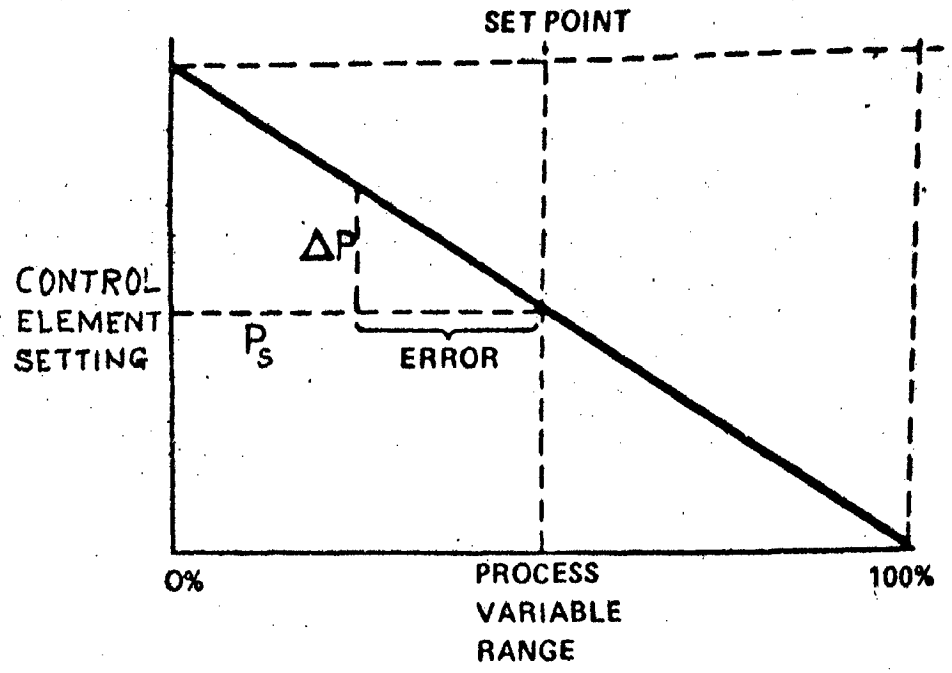


FIG. 5.1.2 PROPORTIONAL CONTROL LAW

P_s = Constant controlled element setting at E=0.

This is the eqn. of a straight line and is shown in Fig. 5.1.2.

Fig. 5.1.3 demonstrates the action of the proportional controller in time. If for some reason, there is a step load disturbance, the controlled element will be adjusted by a proportional amount in the direction necessary to reduce the error to zero. So there will be deviation in P about P_s. But from the control law, it is obvious that P can change only if E does, since P_s is a constant. So because of this deviation, there will be a definite change to the value of E, the error. This residual error, as shown in Fig. 5.1.4 is called the offset. So only for one value of load condition, that is established by the value of P_s, the proportional mode control results in zero error.

5.1.2 INTEGRAL CONTROL

The proportional controller algorithm leads to a steady state error whenever the load exceeds that initially set by the value of P_s; this value must be reset if the error is to be reduced to zero. This can be achieved by a reset mode in which the rate of change of controlled variable is proportional to the error, i.e.

$$\frac{dP}{dt} = K_1 E$$

Take initial condition as zero

$$P = K_1 \int E dt$$

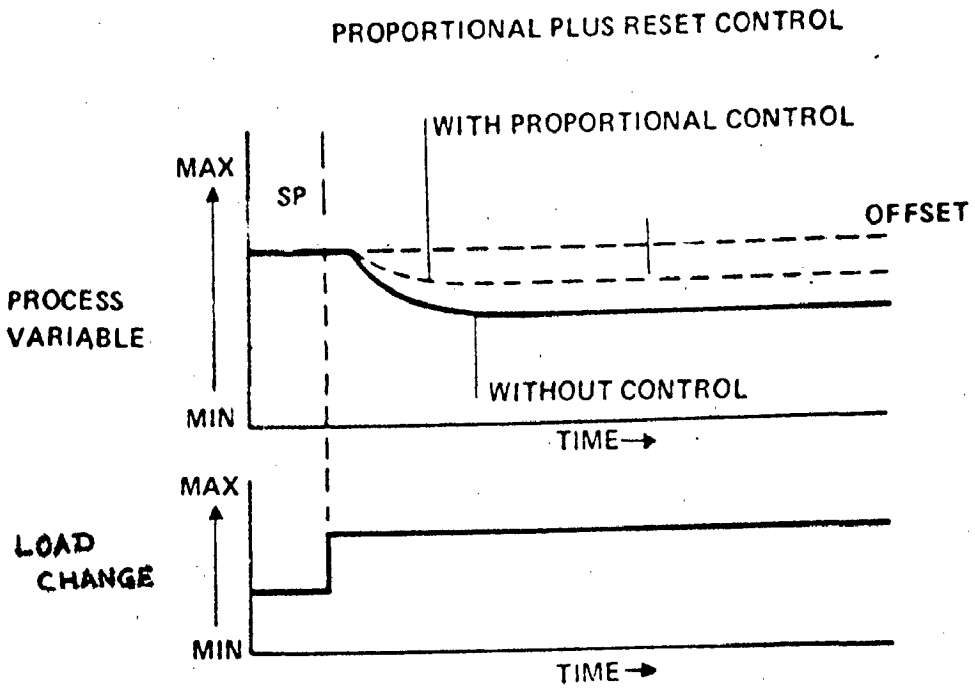
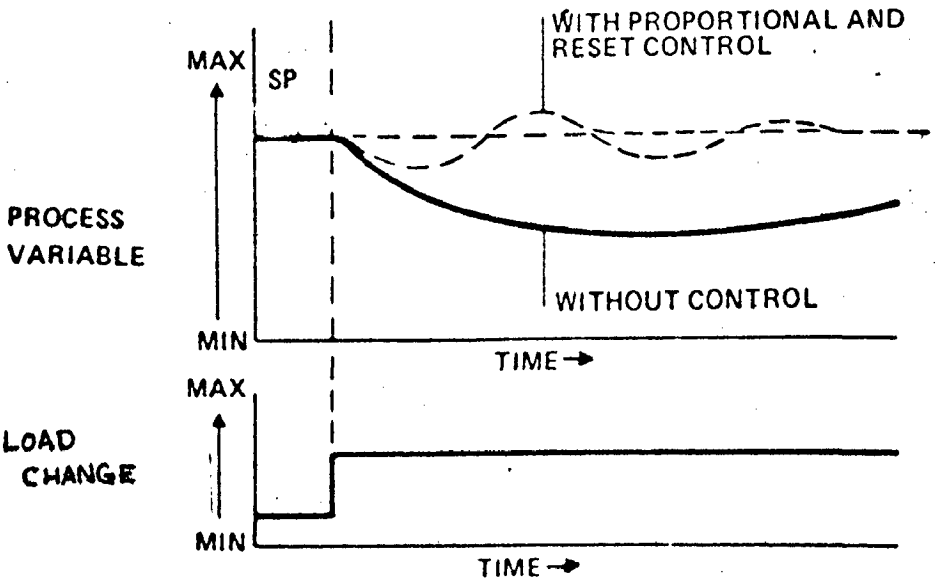


FIG. 5.1.4 Response to load change proportional control only



Response to load change proportional plus reset control

FIGURE 5.1.6 Improvement in closed loop performance due to PI control.

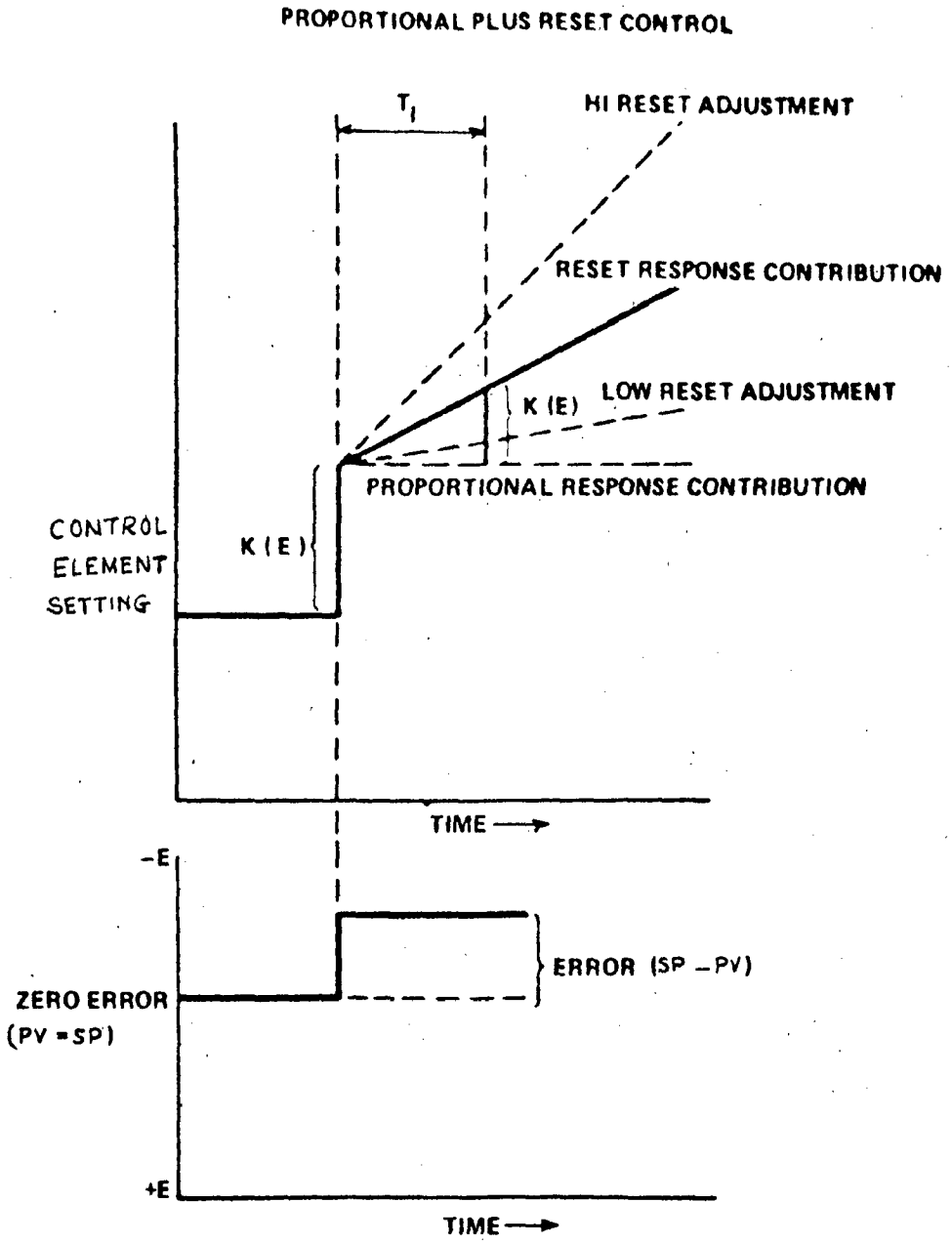


FIGURE 5.1.5 Response to PI control.

This gives the control law for integral control. This will completely null the error and correct the offset, given enough time, since the controller continues to drive the value so long as any error exists.

If both proportional and integral control are combined, the control equation will be

$$P = KE + K_1 \int E\, dt + P_s$$

This is known as PI controller. The behaviour of the PI controller when subject to an error step is shown in Fig. 5.1.5. The controller o/p rises almost instantly by an amount KE as a result of the proportional term. But since the error persists, the integral term adjusts the controlled element at a constant rate (assuming an open loop system), where the slope of rise is given by reset rate K_I. After a time T_I, called the reset time, the controller O/P due to the slope of the integral term becomes equal to the original proportional contribution KE. Thus, PI algorithm in terms of reset time can be expressed as -

$$P = KE + \frac{K}{T_I} \int E\, dt + P_s$$

as

$$K_I = K/T_I$$

Fig. 5.1.6 shows, in terms of a closed loop system, the improvement resulting from the addition of this reset or integral mode. The response of a stable system to a step load change is thus eventually a nulling of the error.

5.1.3 DERIVATIVE (RATE) CONTROL AND THE PID ALGORITHM

As is obvious from Fig. 5.1.6, that a substantial amount of time could elapse before a fairly slow process returns to zero error. A means of improving the controller to have it anticipate the error by sensing when, and how fast, it begins to move can be achieved by sensing the rate of change of error and adding a term proportional to this factor as a correction to the controller. This factor will be

$$P = K_D \frac{dE}{dt}$$

where K_D is the derivative or rate constant.

This rate term affects the controller only during a change in the magnitude of the error. A steady state value of E can be corrected only by the PI algorithms. Combining all these, leads to a three-mode controller, the PID controller. The control law for PID control can be expressed as -

$$P = K\left(E + \frac{1}{T_I} \int E\, dt + T_D \frac{dE}{dt}\right) + P_s$$

The term T_D is the rate or derivative time and is related to K_D, the rate gain constant by

$$K_D = K\, T_D$$

Fig. 5.1.7 demonstrates the performance of a PD-controller. If the set point is moved linearly (ramped) so that the open loop error is a function of time E= ct, the rate action will produce an immediate step change in controlled

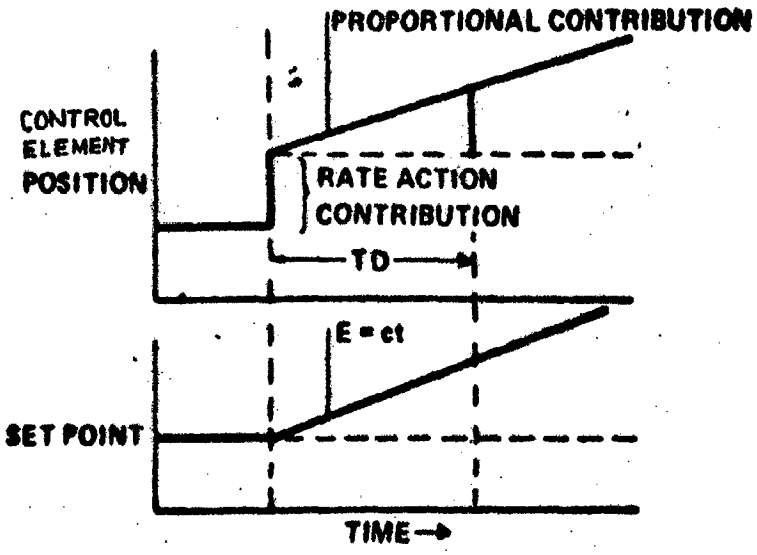


FIGURE 5.1.7 - PD controller action.

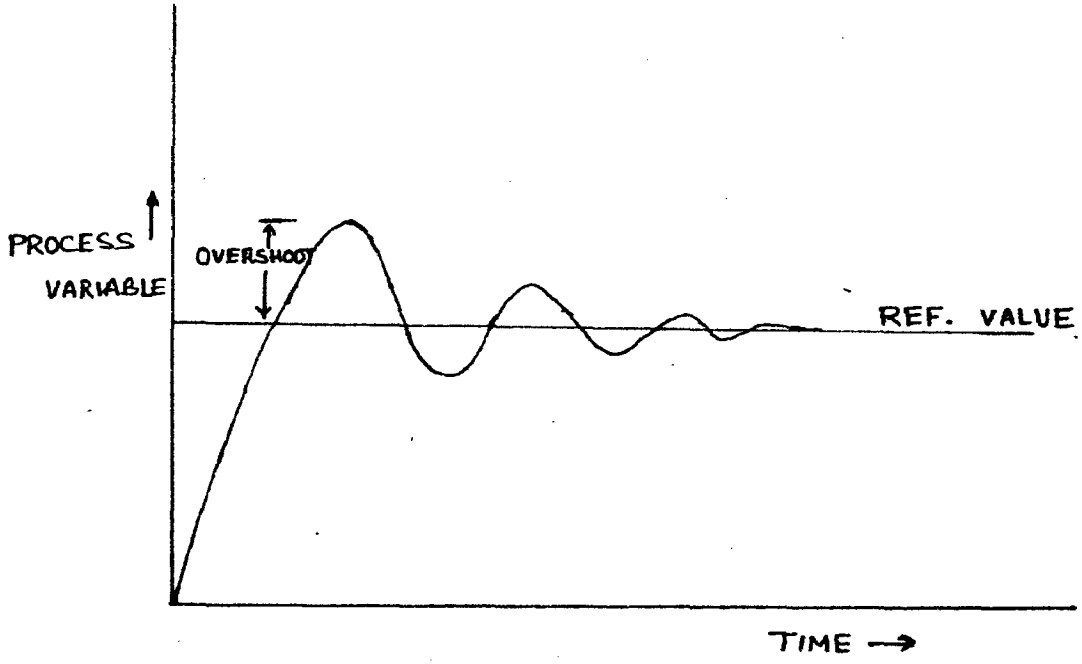


FIG. 5.1.8 : RESPONSE TO PID CONTROL

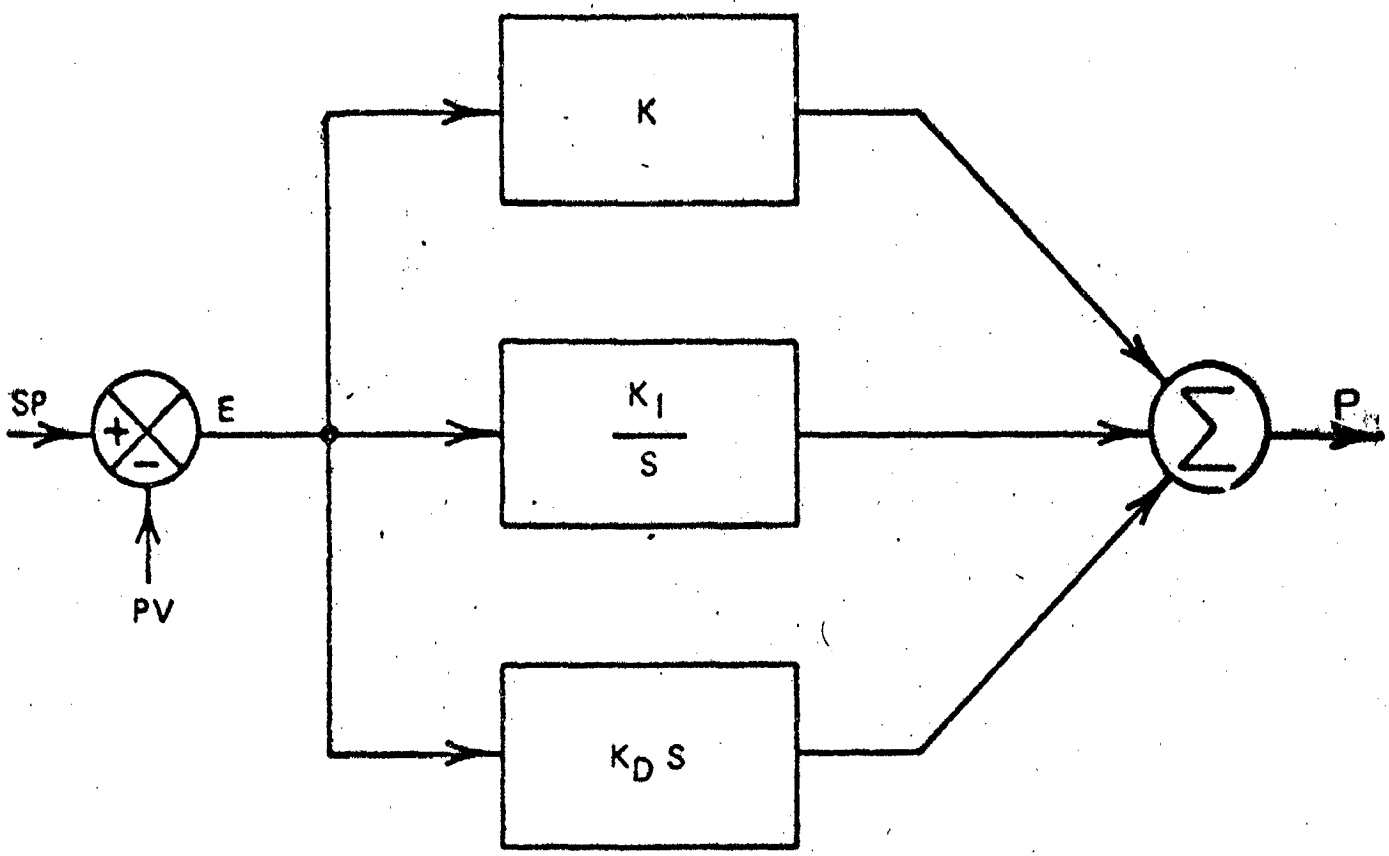


FIGURE 5.1.9 Ideal PID

o/p proportional to the error slope $\dot{e}$. It takes a time T_D for the proportional factor KE to equal this anticipated rate correction.

The response of PID controller is shown in fig.5.1.8.

Modifications of the PID Algorithm

The transfer function of the PID controller is given by $\frac{P}{E} = K\left(1 + \frac{1}{T_I s} + T_D s\right)$, which is shown by fig. 5.1-9.

In order to limit high frequency gain and phase, this is modified by a low pass filter.

$$\therefore \frac{P}{E} = \frac{1}{1+T_F s} K\left(1 + \frac{1}{T_I s} + T_D s\right) \qquad \ldots(5.1.1)$$

If $T_F = \gamma T_2$; $1/T_F$ being low-pass corner frequency

$$K = K_1 \frac{T_1 + T_2}{T_1}$$

$$T_I = T_1 + T_2$$

$$T_D = \frac{T_1 T_2}{T_1 + T_2} \qquad \ldots(5.1.2)$$

then $$\frac{P}{E} = \frac{K_1(1+T_1 s)(1+T_2 s)}{T_1 s (1+\gamma T_2 s)} \qquad \ldots(5.1.3)$$

5.2 SOFTWARE IMPLEMENTATION OF PID CONTROL SCHEME

5.2.1 POSITION ALGORITHM

An algorithm convinent for implementing PID control scheme on microprocessor can be developed by using eqn. 5.1.3, i.e.

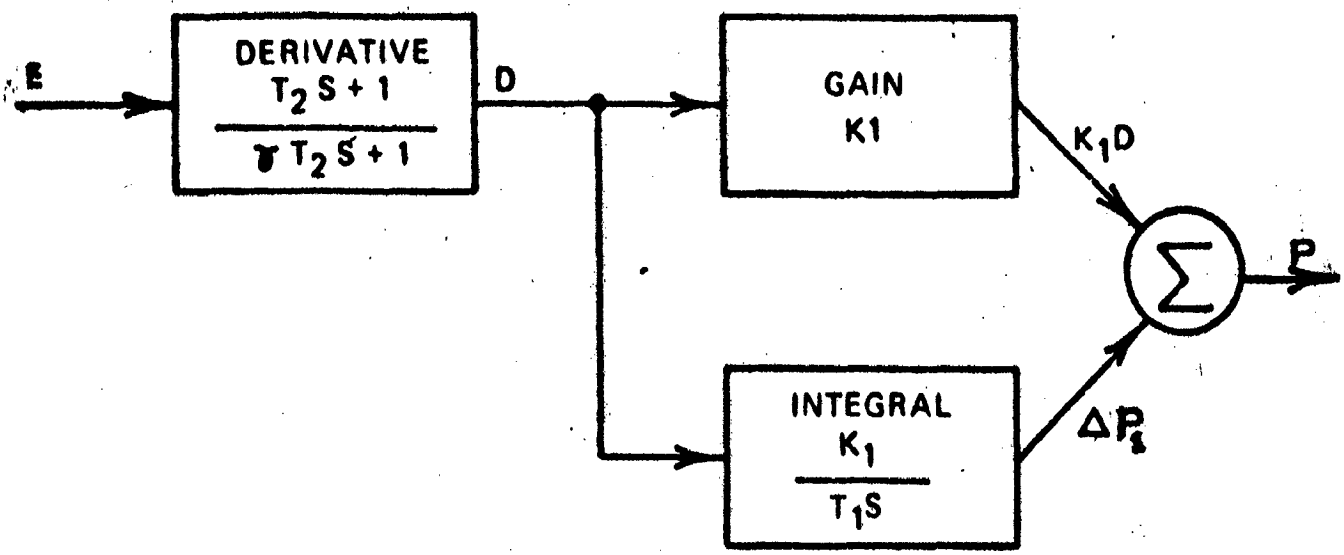


FIGURE 5.2.1 Real PID algorithm block diagram.

$$\frac{P}{E} = \frac{T_2 s+1}{\gamma T_2 s+1} K_1\left(1+\frac{1}{T_1 s}\right)$$

where P = output of the controller

E = error (PV-SP)

T_1 = real integral time constant

T_2 = real derivative time constant

γ = real amplitude constant

K_1 = gain

Fig. 5.2.1 shows the real PID algorithm block diagram. An algorithm can be derived considering each block seperately.

Derivative Block

For derivative block,

$$\frac{D}{E} = \frac{T_2 s+1}{\gamma T_2 s+1}$$

$$\Rightarrow (\gamma T_2 s+1)\, D = (T_2 s+1)E$$

$$\Rightarrow \gamma T_2 \frac{dD}{dt} + D = T_2 \frac{dE}{dt} + E$$

Changing to difference form using

$$\Delta D = D_n - D_{n-1}$$

$$\Delta T = T_n - T_{n-1} = T_s \text{ (sample period)}$$

$$\gamma T_2 \frac{D_n - D_{n-1}}{T_s} + D_n = T_2 \frac{E_n - E_{n-1}}{T_s} + E_n$$

$$\Rightarrow D_n(\gamma T_2 + T_s) = \gamma T_2 D_{n-1} + T_2(E_n - E_{n-1}) + E_n T_s$$

$$D_n = \frac{\gamma T_2}{\gamma T_2 + T_s} D_{n-1} + \frac{T_2}{\gamma T_2 + T_s}(E_n - E_{n-1}) + \left(\frac{T_s}{\gamma T_2 + T_s}\right) E_n$$

$$= D_{n-1} - D_{n-1}\left(1 - \frac{\gamma T_2}{\gamma T_2 + T_s}\right) + \frac{T_2}{\gamma T_2 + T_s}(E_n - E_{n-1}) + \left(\frac{T_s}{\gamma T_2 + T_s}\right) E_n$$

$$= D_{n-1} + \left(\frac{T_2}{\gamma T_2 + T_s}\right)(E_n - E_{n-1}) + \frac{T_s}{\gamma T_2 + T_s}(E_n - D_{n-1}) \quad \ldots(5.2.1)$$

For small values of T_s ($<< T_2$),

$$D_n \simeq D_{n-1} + \frac{1}{\gamma}(E_n - E_{n-1}) + \frac{T_s}{\gamma T_2 + T_s}(E_n - D_{n-1}) \ldots(5.2.2)$$

This is the desired form of the derivative block.

Integral Block

For integral block, the transfer function is

$$\frac{P}{D_n} = \frac{K_1}{T_1 s}$$

$$P_s = \frac{K_1}{T_1} D_n$$

In discrete form,

$$\frac{P_n - P_{n-1}}{T_s} = \frac{K_1}{T_1} D_n$$

$$\therefore \Delta P_I = K_1 \left(\frac{T_s}{T_1}\right) D_n$$

The ratio T_s/T_1 is defined as integral gain constant and K_1 as the system gain.

IDEAL PID

The non-interactive PID algorithm can be written directly by referring to Fig. 5.1-9.

$$P_n = Ke_n + K_I \sum_{i=o}^{n} e_i T_s + K_D \frac{e_n - e_{n-1}}{T_s} + P_s \qquad (5.2.3)$$

Where the summation is the difference equivalent of integration, and

P_s = Average value of controlled variable (offset)

K_I = Integral gain constant, K/T_I

K_D = Derivative constant, KT_D

The advantage of this algorithm is that it gives a fast response.

5.2.2 VELOCITY ALGORITHM

The velocity or incremental form is obtained by subtracting two successive values of P, i.e. $P_n - P_{n-1}$. Solving eqn. (5.2.3) for V_{n-1},

$$P_{n-1} = K e_{n-1} + K_I \sum_{i=o}^{n-1} e_i T_s + K_D \frac{e_{n-1} - e_{n-2}}{T_s} + P_s \qquad (5.2.4)$$

$$\therefore P_n - P_{n-1} = \Delta P_n = K(e_n - e_{n-1}) + K_I e_n T_s + \frac{K_D}{T_s}(e_n - 2e_{n-1} + e_{n-2}) \qquad (5.2.5)$$

One advantage is that the average value P_s has disappeared, meaning that, when the controller is first started, the control loop has not to be initialized by inserting this value manually. In the positional form, if the controlled loop is switched from manual to automatic control, the process will bump unless the controller is aligned with the present controlled element position. The velocity algorithm is

bumpless. Another advantage is that the elimination of the summation eliminates the danger of windup, a condition in which the controller saturates its integral term when for some reason an error signal persists.

5.2.3 IMPROVED DERIVATIVE AND INTEGRAL COMPUTATION

The PID algorithm can be modified further. In the first differences, taken for derivative terms in position algorithm and even for second differences in velocity algorithm, noise may be a problem.

The simple first difference is

$$\Delta e_n = e_n - e_{n-1}$$

and the second difference,

$$\begin{aligned}\Delta^2 e &= \Delta e_n - \Delta e_{n-1} \\ &= (e_n - e_{n-1}) - (e_{n-1} - e_{n-2}) \\ &= e_n - 2e_{n-1} + e_{n-2}\end{aligned}$$

Employing four point central difference technique of interpolation for velocity algorithm.

$$\text{Let } e' = \frac{e_n + e_{n-1} \quad e_{n-2} + e_{n-3}}{4}$$

where e_n to e_{n-3} are equally spaced at the sampling interval.

$$\frac{\Delta e}{T_s} = \left(\frac{e_n - e'}{1.5T_s} \right) + \frac{e_{n-1} - e'}{0.5T_s} + \frac{e' - e_{n-2}}{0.5\,T_s} + \frac{e' - e_{n-3}}{1.5\,Ts}\Big)/4$$

$$= \frac{1}{6Ts} (e_n - e_{n-3} + 3e_{n-1} - 3e_{n-2})$$

Similarly integral term for $\sum_{i=o}^{n} e_i$ can be substituted by trapezoidal rule

$$\sum_{i=o}^{n} \frac{e_i + e_{i-1}}{2}$$

So velocity algorithm can be modified as

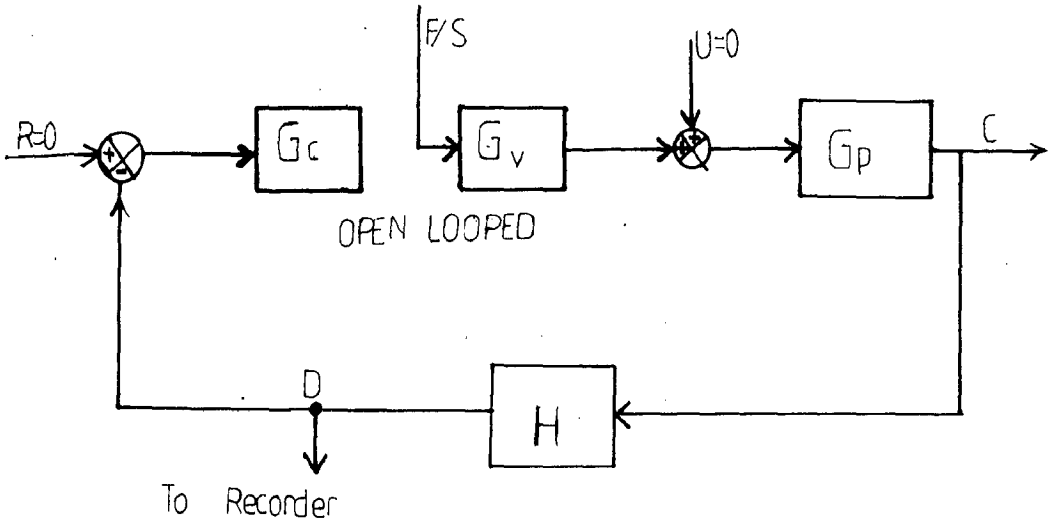
$$\Delta P_n = K(e_n - e_{n-1}) + \frac{K_D}{6T_s}(e_n + 2e_{n-1} - 6e_{n-2} + 2e_{n-3} + e_{n-4}) + K_I\left(\frac{e_n + e_{n-1}}{2}\right)T_s \qquad (5.2.6)$$

However, for the present work, the position algorithm is taken and is approximated by taking 15 errors only for the integral term. Thus,

$$P_n = K\, e_n + K_I \sum_{i=n-14}^{n} e_i T_s + K_D \frac{e_n - e_{n-1}}{T_s} + P_s \qquad (5.2.7)$$

5.3 ESTIMATION OF CONTROL PARAMETERS

It is always possible to evaluate control parameters by deriving a theoritical model for the process to be controlled. However, the initial controller settings can also be evaluated from the experimental data by plotting the reaction curve. Process reaction curve method, given by Cohen and Coon [9] is one of such methods. This method is described below.



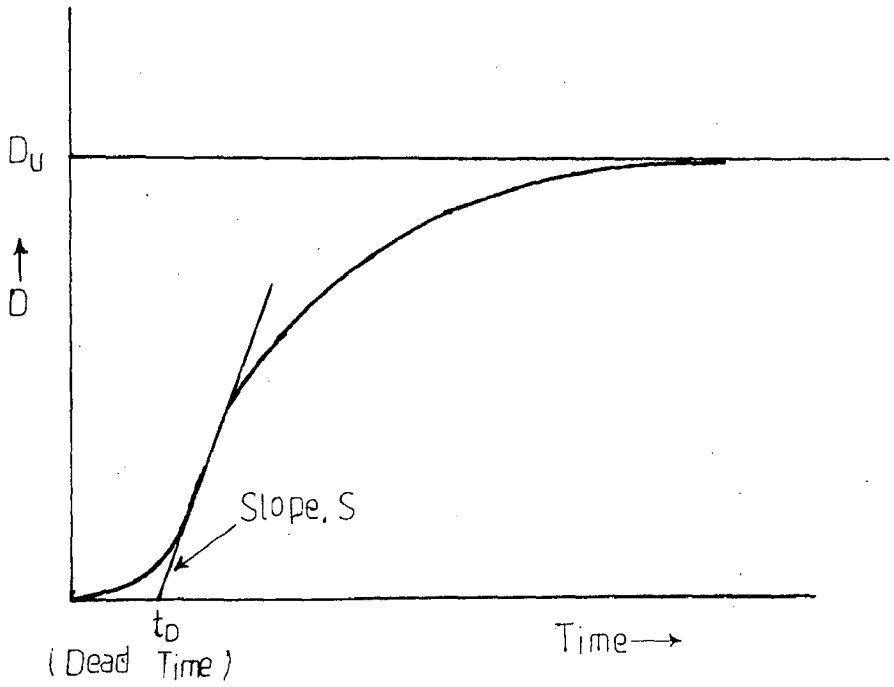


FIG. 5.3.1 – PROCESS REACTION CURVE WITH BLOCK DIG. TO MEASURE IT.

PROCESS REACTION CURVE METHOD

A block diagram for measurement of process reaction curve and the process reaction curve are shown in Fig. 5.3.1.

This method consists of applying a small step change in the manipulated variable to the opened control loop and recording the curve of measured variable versus time, called the process reaction curve. It is assumed that no load changes occur during the test. In addition, all the dynamic components of the loop other than the controller must be included between the point of application of the manipulated variable change and the point of recording the response.

A tangent is drawn to the reaction curve at the point of inflection. The intercept of this tangent on the abscissa is taken as the apparent dead time T_d. The slope of the tangent, S, is proportional to 1/T, the reciprocal of the apparent time constant. In this case D_u is the ultimate response. Hence,

$$T = D_u/S$$

The steady state gain between F and D is calculated as

$$K_p = D_u/F$$

Using the values of T_d, T and K_p determined in this manner, the following controller settings are recommended by Cohen and Coon .

Proportional :

$$K = \frac{1}{K_p} \frac{T}{T_d} \left(1 + \frac{T_d}{3T}\right)$$

Proportional -Integral :

$$K = \frac{1}{K_p} \frac{T}{T_d} \left(\frac{9}{10} + \frac{T_d}{12T}\right)$$

$$T_I = T_d \left(\frac{30 + 3T_d/T}{9+20T_d/T}\right)$$

Proportional- Derivative :

$$K = \frac{1}{K_p} \frac{T}{T_d} \left(\frac{5}{4} + \frac{T_d}{6T}\right)$$

$$T_D = T_d \frac{6-2T_d/T}{22+3T_d/T}$$

Proportional- Integral- Derivative :

$$K = \frac{1}{K_p} \frac{T}{T_d} \left(\frac{4}{3} + \frac{T_d}{4T}\right)$$

$$T_I = T_d \frac{32+6T_d/T}{13+8T_d/T}$$

$$T_D = T_d \frac{4}{11+2T_d/T}$$

These relations give a good estimate of controller settings and are used to estimate the control parameter values when the system has a dead time. However, this method is not used for the present work.

CHAPTER VI

SOFTWARE DEVELOPMENT

The philosophy of the CSTR control is discussed in earlier chapters. This requires the development of some routines, necessary to implement PID control scheme by software on the CSTR. A top-down approach is taken for this purpose i.e. main program written first and the necessary routines next. The software tree for the control scheme is shown in Fig. 6.1.1.

The whole software is grouped into five main parts as given below:

1) Main Program
2) Serial Communication and Functional Commands
3) Floating Point Arithmetic
4) Routines for Various Interfaces
5) Temperature monitoring and other routines.

However, the floating point arithmetic routines are not used for controlling the CSTR. These may be used for a precise control and are therefore developed.

A brief description of all these routines along with the flow charts wherever necessary is given below. The whole program listing is given at the end of this chapter.

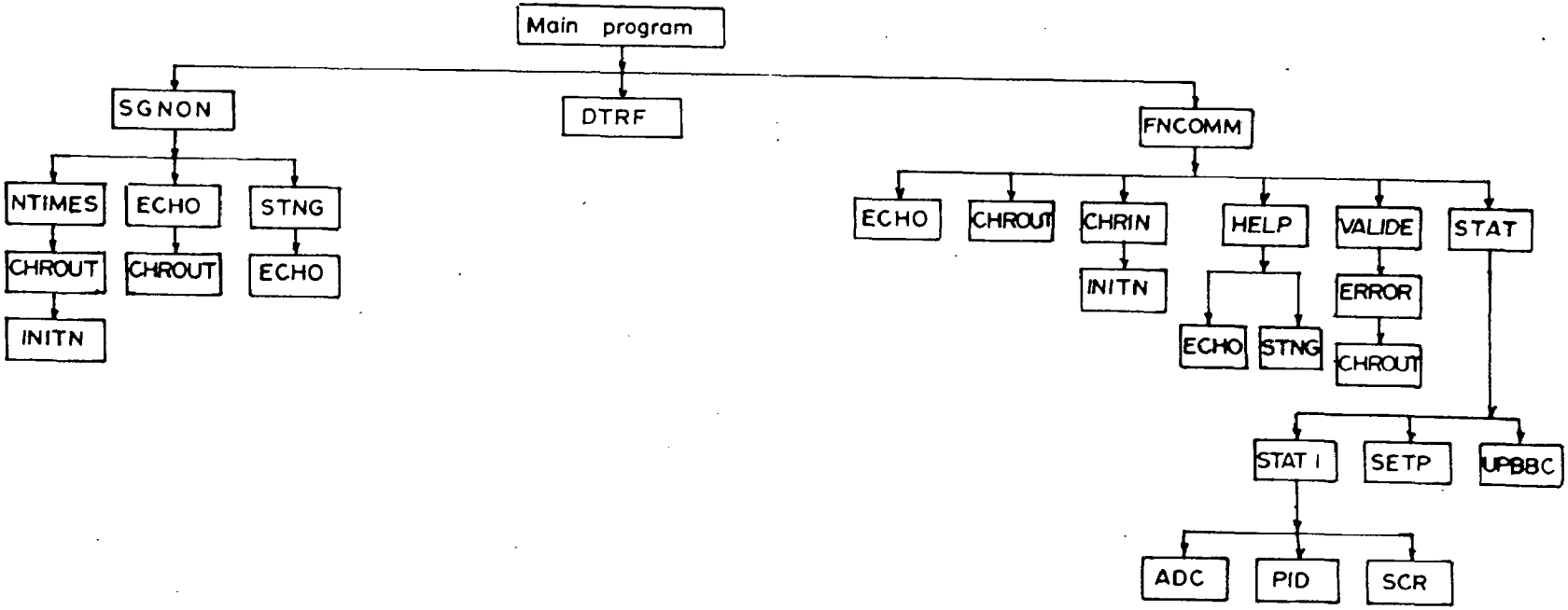


FIG. 6.1.1 SOFTWARE TREE FOR CSTR CONTROL (TOP-DOWN)

6.1 MAIN PROGRAM

The main program developed for temperature control of the CSTR first transfers the necessary data from EPROM to the RAM area. This is done for convinience so that the data is not to be entered in RAM, every time the power is put on. This data transfer includes the addresses of type 252 and type 253 interrupts. (from F000:3200-3207 to 0000: 03F0-03F7) which are used by the routines FREQ and STAT1 respectively. Also it includes the values of the PID parameters K, $K_I T_S$ and K_D/T_S (from F000:3208-320D to 0000: 3F1E-3F23), necessary for the routine PID. Moreover, for the same routine, all the errors in locations 0000:3F00-3F1D are set to zero. All the data transfer is done by routine DTRF.

Then it displays a sign-on message, the ASCII Codes of characters of which are stored in locations F000:3000-3157 by the routine SGNON. The FNCOMM routine then expects a valid functional command to be given and executes it. This routine, as discussed later assumes certain data to be stored in locations F000:3170-3189, 31A0-31A7, 3F00-3F03. If the functional command is STAT, it asks for the ref. temp in BCD and stores the value in unpacked BCD format in memory locations 0000:3EF0-3EF1. This routine requires certain data to be stored in

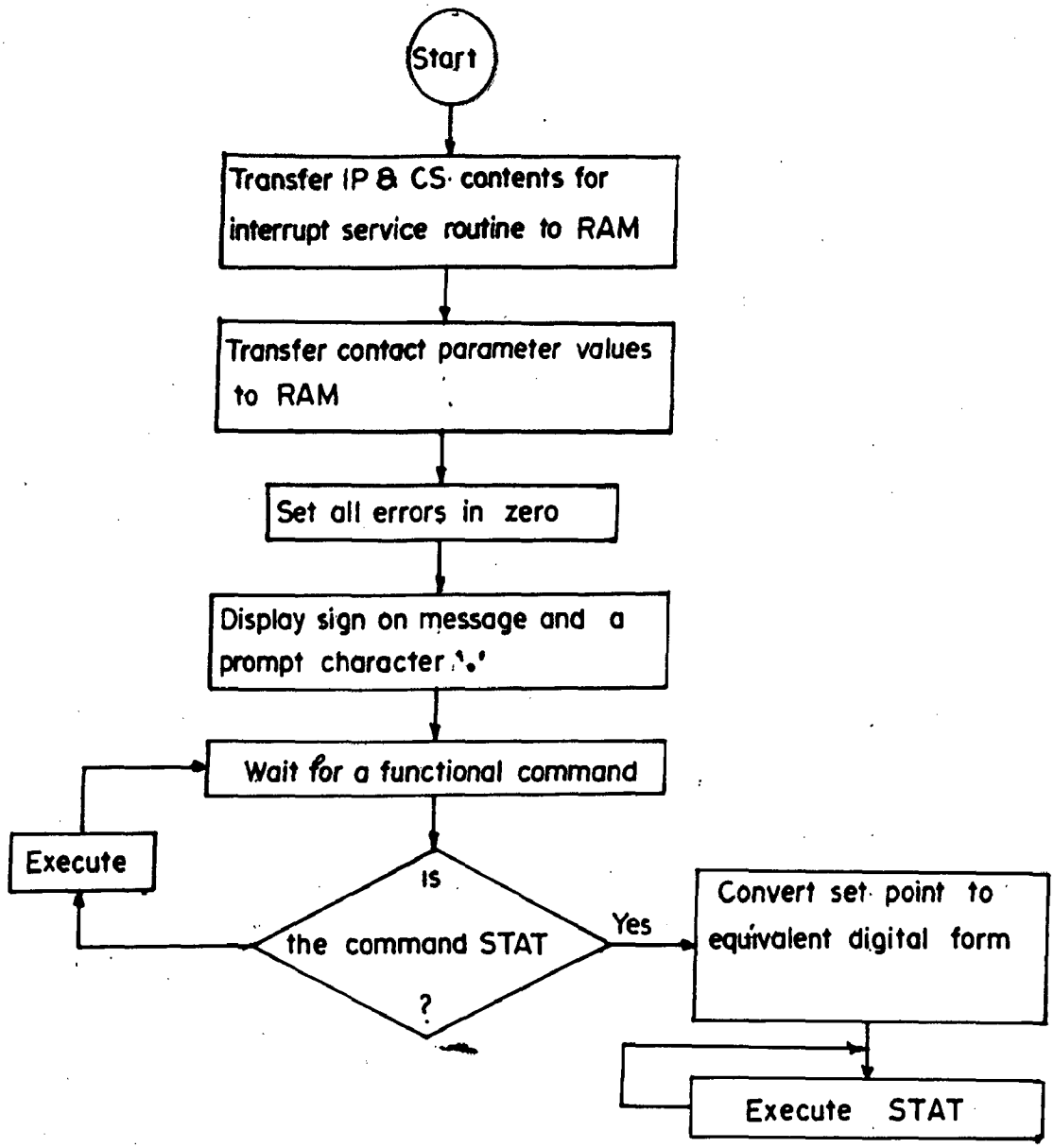


FIG. 6·1·2 FLOW CHART FOR MAIN PROGRAM.

locations F000:31B0-31B9. The routine UPBBC converts the unpacked BCD ref. temp. value into an equivalent digital word and stores it at location 0000:3EF0-3EF1 for using it by the routine STAT1.

The main program starts at an address F000:0000.

6.2 SERIAL COMMUNICATION AND FUNCTIONAL COMMANDS

These routines are developed for man-m/c communication through CRT. They consist of initializing 8251 through which the serial communication is made, inputting a character's ASCII code from the ASCII keyboard, outputting a character or a no. of characters to the console, echoing a character given from the ASCII keyboard, sending sign-on message and executing the various functional commands given through CRT.

These routines are given below with the flow charts wherever necessary.

1) SUBROUTINE INITN:

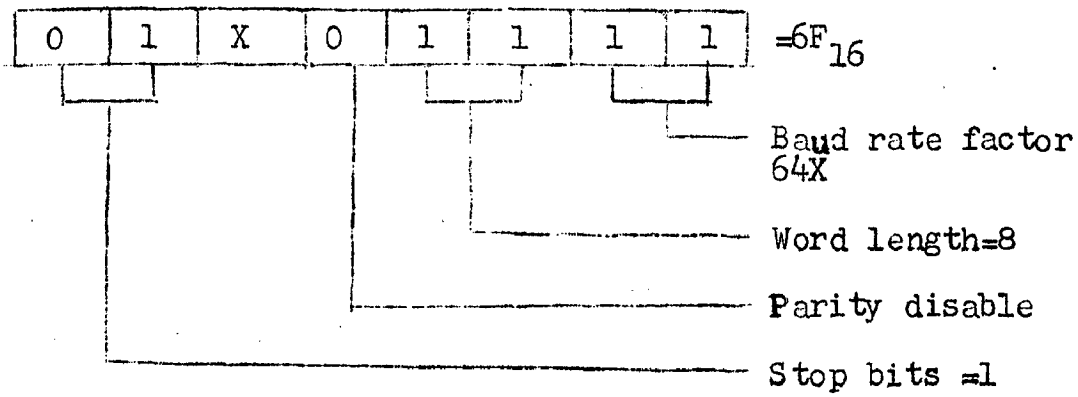
This routine initializes the 8251 for

Clock frequency = 64 x band rate,

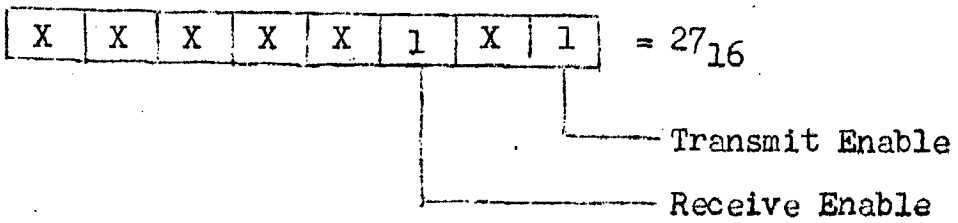
Stop bit = 1

Word length = 8

No parity.



Command Instruction Format



All the routines developed for serial communication and functional commands assume that this routine is already called.

Calling address - F000:1000.

2) SUBROUTINE CHRIN:

This routine inputs the ASCII code of a character pressed on the ASCII keyboard in the reg. AL.

Calling address - F000:1010

3) SUBROUTINE CHROUT:

This routine outputs a character whose ASCII code is in AL to the console.

Calling address - F000: 1020.

4) SUBROUTINE ECHO:

This routine echoes a character whose ASCII code is in(AL) to the console.

'ESC' is echoed as '$'

'CR' is echoed as 'LF' + 'CR'.

Calling address -F000:1040.

5) SUBROUTINE NTIMES:

This routine sends a character whose ASCII code is in reg. AL to the console as many times as the contents of reg. CL. If (CL)=00, the character is outputted 256 times.

Calling address- F000: 1060

6) SUBROUTINE STNG:

This routine sends an string of characters whose ASCII codes are in memory locations starting from (DS):(SI). The no. of characters are given by (CL). If (CL)=00 the string is assumed to have 256 characters.

Calling address - F000: 1070.

7) SUBROUTINE ERROR

This routine sends '*' to the console as an error message.

Calling address - F000:1080.

8) SUBROUTINE SGNON:

This routine clears the whole CRT screen and displays the sign-on message -

CSTR

READY

GUIDED BY 1) SH. M.K.VASANTHA
2) Sh. B.MOHANTY

The ASCII codes of all these characters are stored in memory locations F000: 3000 - 3157. A data FF in any of these locations indicates the no. of blanks given in the next location. The characters of 'CSTR READY' are displayed by a (4)x (7) matrix formed by the character '#' e.g. character 'C' is displayed as

```
# # # #
#
#
# # # #
```

Calling address- F000:1090

9) SUBROUTINE FNCOMM

This routine displays a prompt character '.', indicating that it is expecting any command to be given through the CRT. If a valid command is given, the routine written for that command is executed and after execution a prompt character is again displayed expecting a new command to be given. The ASCII codes of the characters given in functional command are stored in a BUFFER RAM area starting from 3EA0 , until 'CR' is pressed. The characters in the buffer can be deleted

in the reverse sequence they are entered till a character '/' is pressed. Pressing the character '/' when there is nothing in the buffer or giving an invalid command are indicated by displaying a message -

* 'HELP' FOR VALID COMMANDS

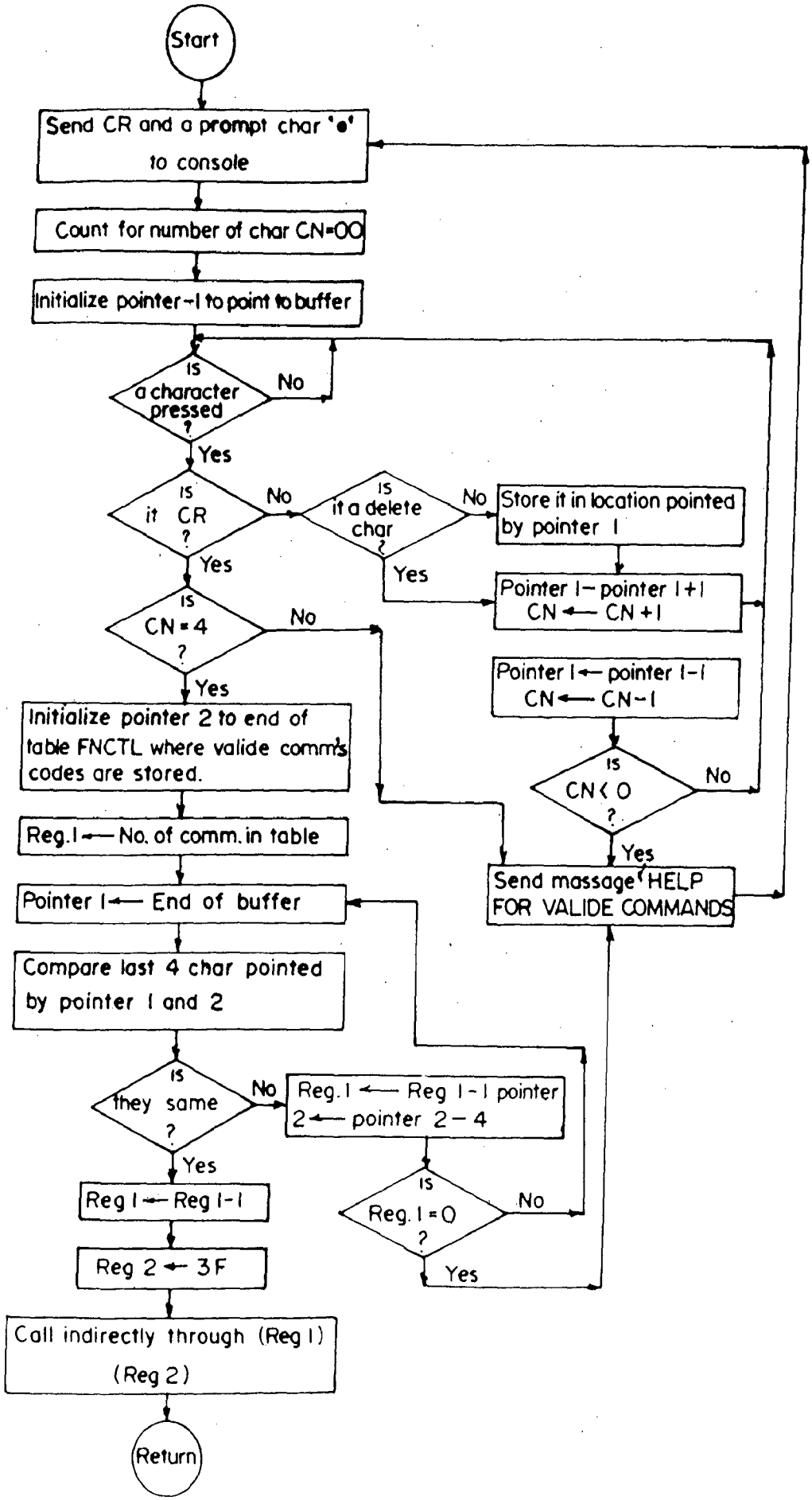
and further it waits for the next command. Each commands is assumed to have 4 characters. When 'HELP' is given , it displays the list of valid commands which include HELP, and STAT.STAT starts the control program execution by first executing SETP, which asks for the set point (ref.temp) and stores it in the mem. location 0000:3EE0-3EE1 after converting into the digital equivalent. The ref. temp. is taken as 40°C if nothing is entered.

STAT continuously executes the control program. To change the set point when this program's execution is going on, the key RESET on Hex-key board of the kit is to be pressed, and then start all over again.

The addresses to serve the various functional commands are stored in memory location 3F00 onwords.

A no. of more functional commands can be developed by making small changes in this routine. These changes are given below:

i) Change the value of NFC in the instruction MOV AL,NFC located at F000:1131 to the required no. of functional commands.


Start
Send CR and a prompt char '•' to console
Count for number of char CN=00
Initialize pointer-1 to point to buffer
IS a character pressed ?
No
Yes
IS it CR ?
No
IS it a delete char ?
No
Store it in location pointed by pointer 1
Yes
Pointer 1← pointer 1+1
CN ← CN+1
Yes
IS CN = 4 ?
No
Pointer 1← pointer 1-1
CN ← CN-1
IS CN < 0 ?
No
Yes
Initialize pointer 2 to end of table FNCTL where valide comm's codes are stored.
Reg.1 ← No. of comm. in table
Pointer 1← End of buffer
Send massage 'HELP FOR VALIDE COMMANDS'
Compare last 4 char pointed by pointer 1 and 2
IS they same ?
No
Reg.1 ← Reg 1-1 pointer 2 ← pointer 2-4
IS Reg.1 = 0 ?
No
Yes
Yes
Reg 1 ← Reg 1-1
Reg 2 ← 3F
Call indirectly through (Reg 1) (Reg 2)
Return

ii) Store the ASCII codes of characters in the new commands F000:31A8 onwards in table FNCTL in continuation to those of previous commands. Each command is assumed to have four characters.

iii) Store the 2 byte addresses of each functional command routine F000: 3F04 onwards in continuation to those for previous commands.

The flow chart for this routine is shown in Fig. 6.2.1

Calling address - F000:10E0.

10) SUBROUTINE HELP

This routine displays the list of valid functional commands as -

1) HELP

2) STAT

Calling address - F000:1180

11) SUBROUTINE VALID

This routine checks if a character whose ASCII code is in(AL), a valid BCD no. (i.e. 0 to 9) or not. If not, it sends '*' as an error message to the console. If the character is a valid BCD no., it is converted into unpacked BCD form by subtracting 30_{10} from the character's ASCII code.

Calling address - F000:11B0

12) SUBROUTINE SETP:

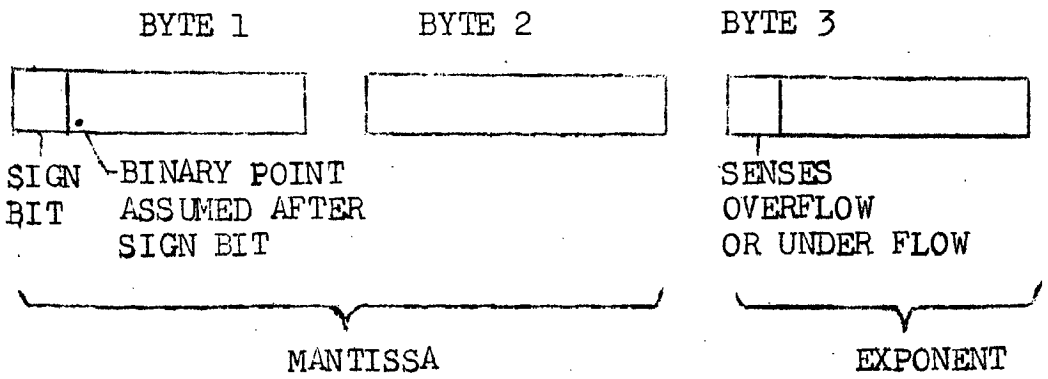
This routine asks for the REF.TEMP and waits for the BCD value to be given. If any digit is not a valid BCD digit, an error message '*' is displayed and the digit is not accepted. The last two valid BCD digits entered are accepted until a 'CR' is pressed. If nothing is entered, the set point is taken as 40°C. Finally, the set point in unpacked BCD format is stored in mem. locations 0000:3EF0-3EF1.

Calling address - F000:11C0

6.3 FLOATING POINT ARITHMETIC

FLOATING POINT NUMBER REPRESENTATION:

The floating no. consists of two parts- 15 -bit mantissa and 7-bit exponent. So it can be presented by three consecutive bytes as



2's complement form has been used. The leftmost bit (16th bit) of mantissa is used as a sign bit. The left most bit of exponent is used for sensing the overflow/underflow condition of the floating point no. The remaining 7-bits of the exponent

could express 0 through 127. However, to express negative exponents, the no. 64_{10} or 40H (offset) has been added to the desired exponent (excess 64 form).

All the numbers are assumed to be in the normalised form, i.e., the range of mantissa is taken from ± 0.5 to 1. With this representation, the range of possible floating point numbers is ± (0.27105×10^{-19} to 0.92231×10^{19}). The accuracy of representation is 1 part in 2^{15} (approx 0.003 %). The floating point +ve and -ve overflow are 0.92231×10^{19} ($7FFF_{16}$ $7F_{16}$) and -0.92231×10^{19} (8001_{16} $7F_{16}$) respectively. Underflows are set equal to zero.

All the numbers are assumed to be in the normalised form, i.e., the range of mantissa is taken from ± 0.5 to 1. With this representation, the range of possible floating point numbers is ± (0.27105×10^{-19} to 0.92231×10^{19}). The accuracy of representation is 1 part in 2^{15} (approx. 0.003 %). The floating point +ve s-ve overflow are 0.92231×10^{19} ($7FFF_{16}$) and -0.92231×10^{19} (8001_{16} $7F_{16}$) respectively. Underflows are set equal to zero.

All the floating point arithmetic routines developed assume that the mantissas of operand 1 and operand 2 are in registers. AX and DX and their exponents in registers BL and BH respectively. Thus, a representation for operand 1 is taken as (AX)(BL) and that for operand 2 as (DX)(BH).

The routines developed for floating point arithmetic include the basic arithmetic operations- Integer to Floating

Point and Floating point to Integer conversion, Normalisation, Addition, Subtraction, Multiplication and Division.

1. SUBROUTINE INTF

This routine converts an integer in reg. (AX) into a floating point no. with its mantissa in (AX) and exponent in (BL). The flow chart for this routine is shown in Fig.6.3.1.
Calling address - F000: 1300.

2. SUBROUTINE FTIN:

This routine converts a floating point no. in (AX)(BL) into an integer in (AX). The integer is set to $7FFF_{16}$ or 8001_{16} depending on whether the floating point no. is +ve or -ve respectively, when the result exceeds 16 bits. The result is set to 0000 when it is less than 0001. The flow chart for this routine is shown in Fig. 6.3.2.
Calling Address - F0000: 1350

3. SUBROUTINE NORMA:

This routine converts a floating point no. in (AX)(BL) into a nonmalised floating point no. in (AX)(BL). The flow chart for this routine is shown in Fig.6.3.3.
Calling Address - F000: 1380

4. SUBROUTINE SUBTRN/ADDN:

The routine SUBTRN subtracts operand 1 in (AX)(BL) from the operand 2 in (DX)(BH) by adding the later to the 2's complement of the former through the routine ADDN. The routine

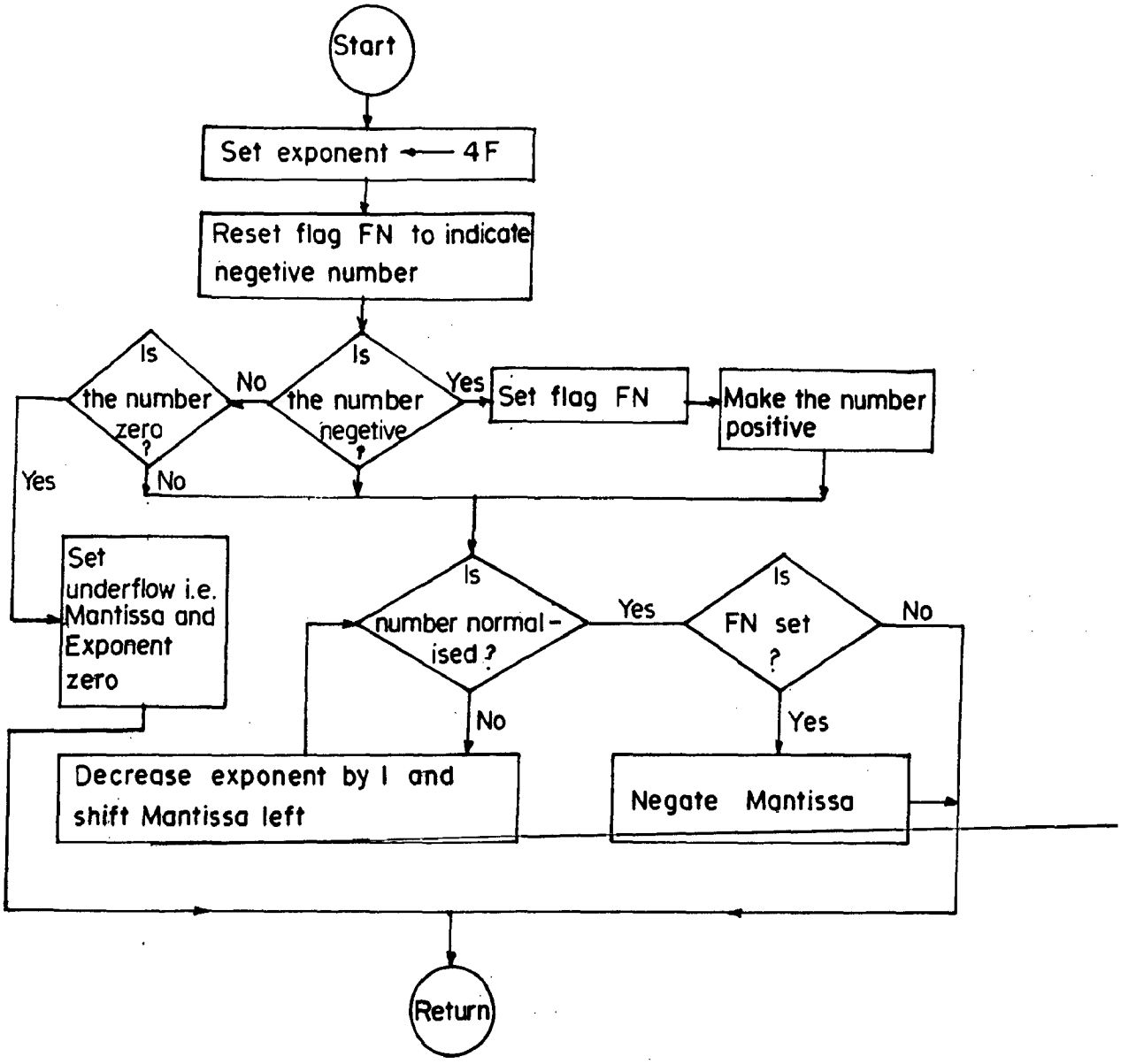


FIG. 6·3·1 FLOW CHART FOR SUBROUTINE INFT.

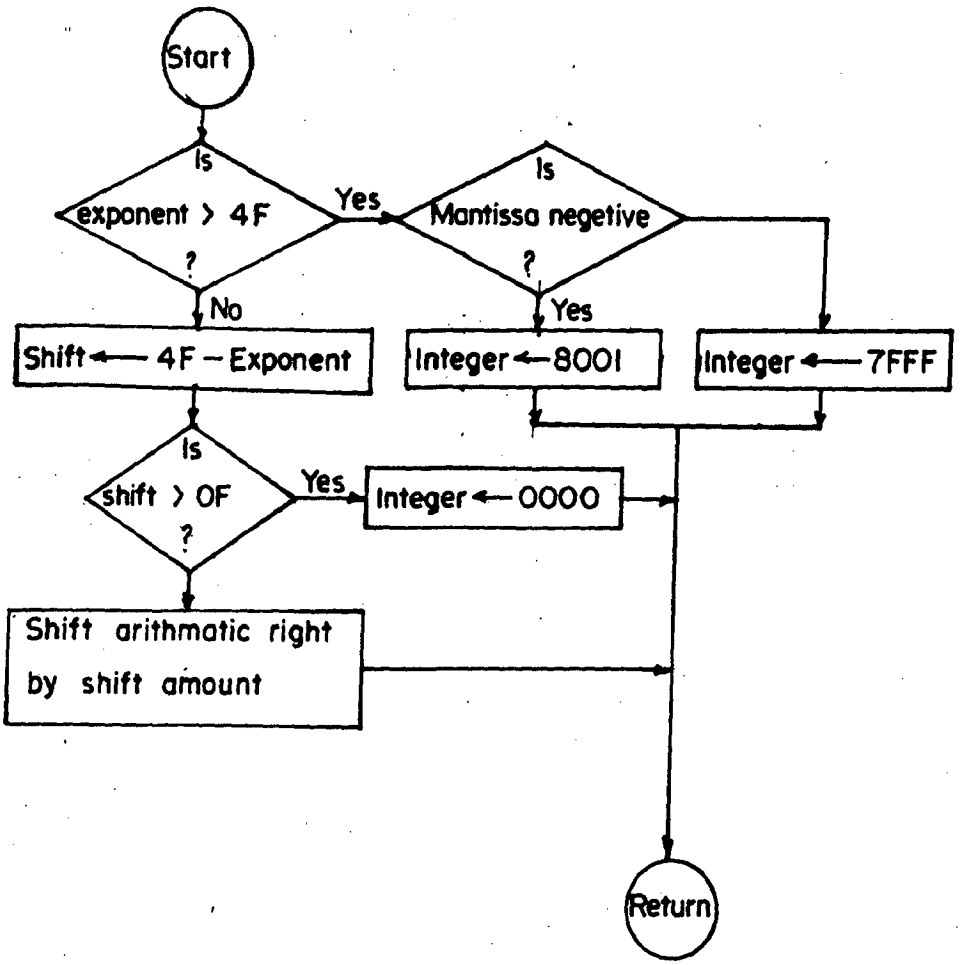


FIG. 6·3·2 FLOW CHART FOR SUBROUTINE FTIN.

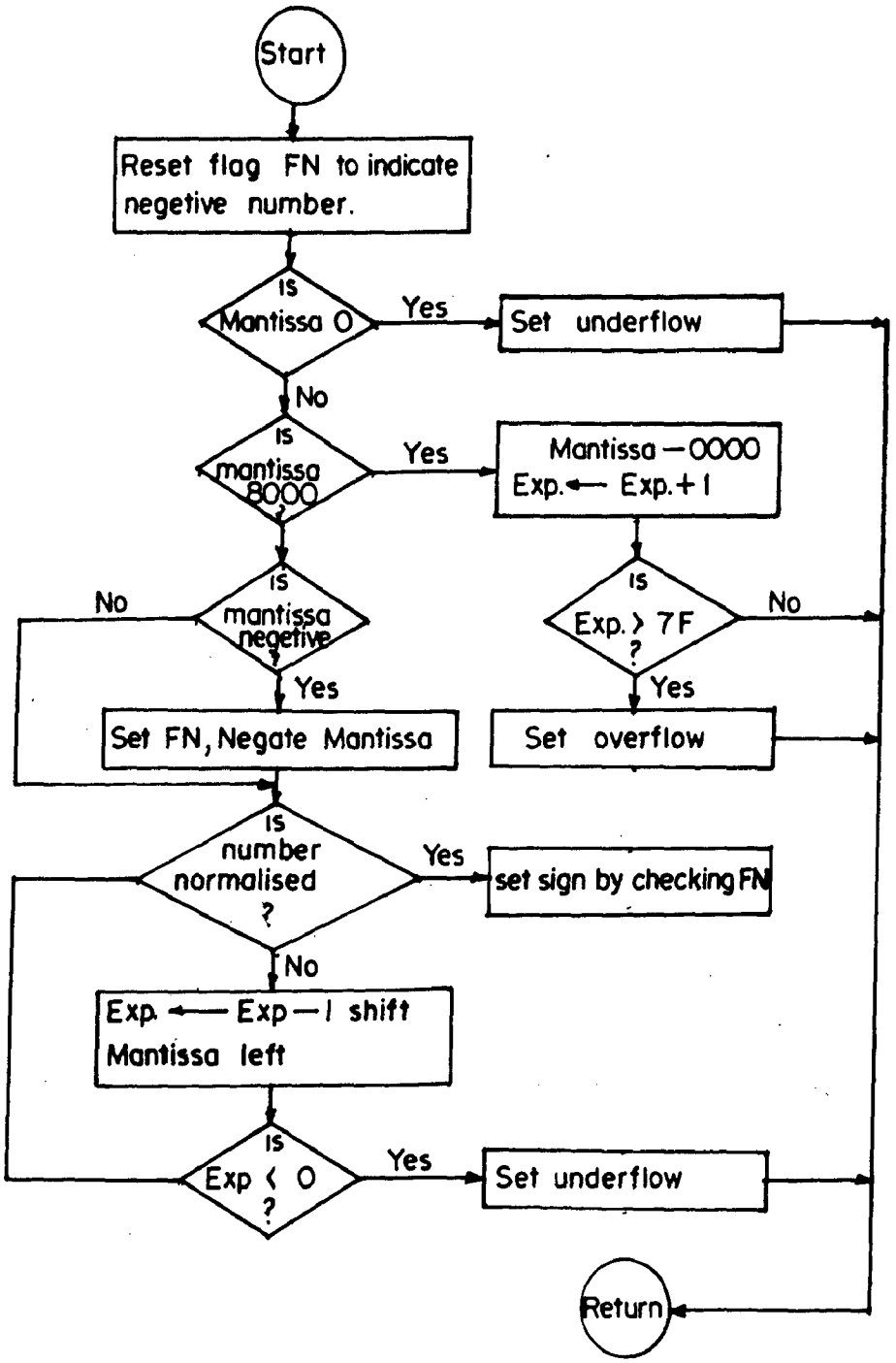


FIG. 6·3·3 FLOW CHART FOR SUBROUTINE NORMA.

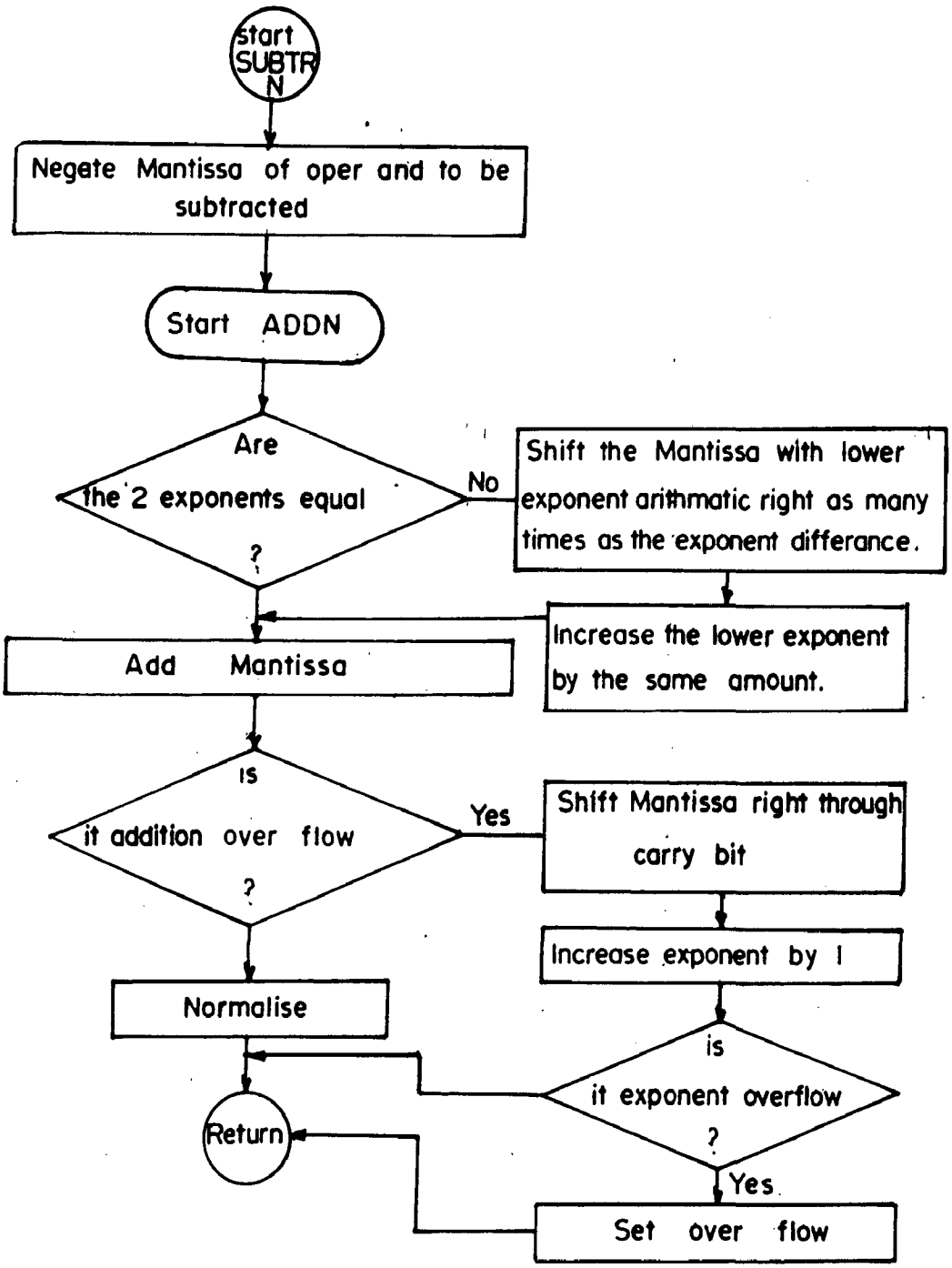


FIG. 6·3·4: FLOW CHART FOR SUBROUTINE SUBTRN/ADDN.

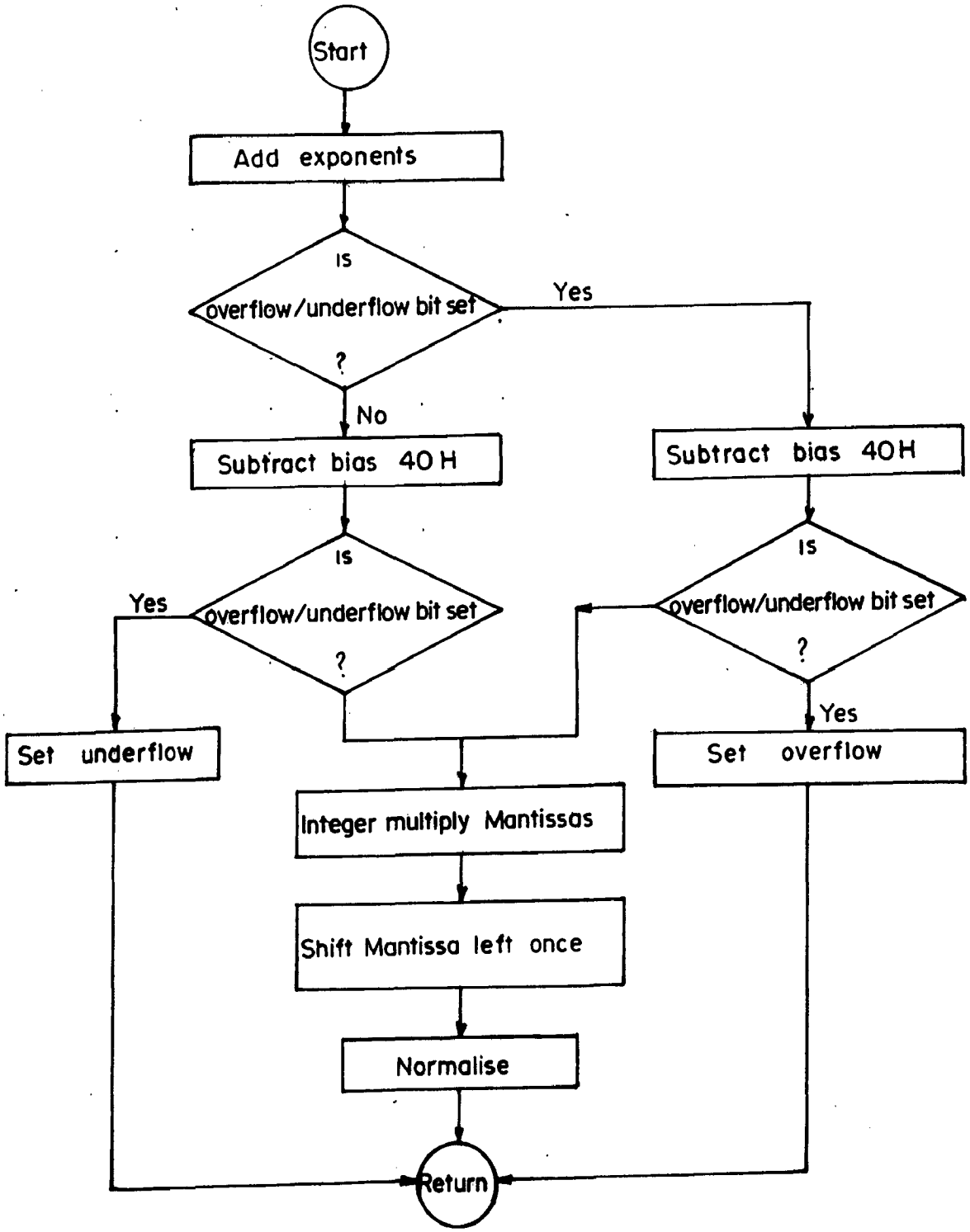


FIG. 6·3·5 FLOW CHART FOR SUBROUTINE MULT.

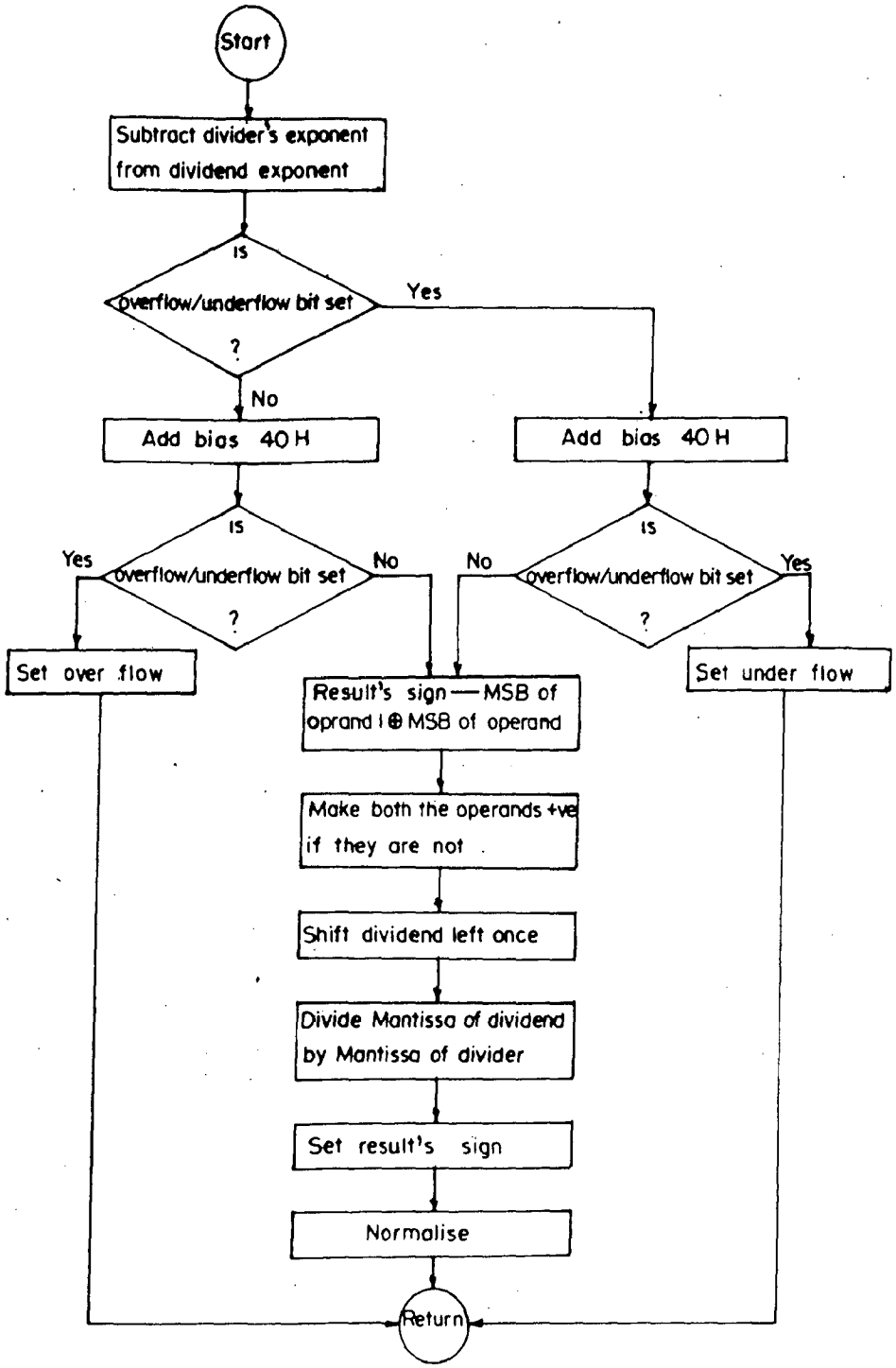


FIG. 6.3.6 FLOW CHART FOR SUBROUTINE DIV.

ADDN adds the two operands. The result of both the operations are returned in (AX)(BL).

This routine has two entries, one for SUBTRN and another for ADDN. The entries for SUBTRN and ADDN are at memory locations F000:13D0 and 13D2 respectively.

The flow chart for these routines is given in Fig. 6.3.4.

5. SUBROUTINE MULT :

This routine multiplies the two normalised operands in (AX)(BL) and (DX)(BH) and returns the normalised result in (AX)(BL).

Calling Address - F000:1410

6. SUBROUTINE DIV:

This routine divides a normalised operand in (DX)(BH) by a normalised operand in (AX)(BL) and returns the normalised result in (AX)(BL).

Calling Address - F000:1450

6.4 ROUTINES FOR VARIOUS INTERFACES :

These routines are developed for initiating and governing the functions of the various hardware interfaces developed. These include the routines for driving stepper motor, issuing firing pulses to SCR converter bridge, initiating ADC and inputting digital equivalent of the voltage applied at I/P of ADC, counting the frequency of the pulses outputted by C-F converter of level transducer. These routines are described below.

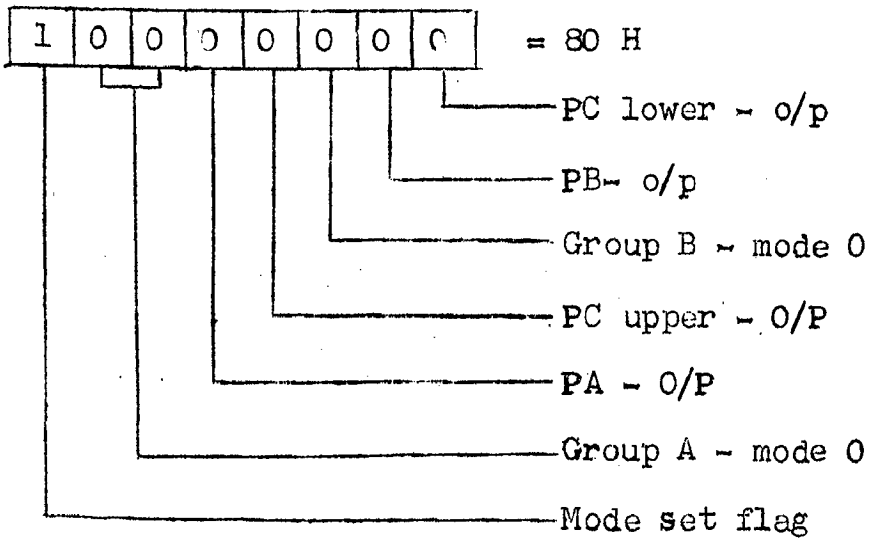
1) SUBROUTINE STPM :

This routine issues pulses to the stepper motor driver circuit to rotate the stepper motor through 1.8°. The pulses are issued from 8255-I, port A. The duration of the pulses is decided by a DELAY subroutine, which is fixed to about 30 ms.

This routine rotates the motor in one direction. To rotate it in opposite direction, as discussed earlier in stepper motor driving circuit, the pulses are to be issued in the reverse sequence.

This routine could be used to control level of water in the tank by controlling the position of value at the outlet of the tank, which in turn is decided by how much to rotate the stepper motor. For the present work, this routine is not being used.

All the ports of 8255-I are taken as simple o/p ports. The mode word for initializing 8255-I is taken as

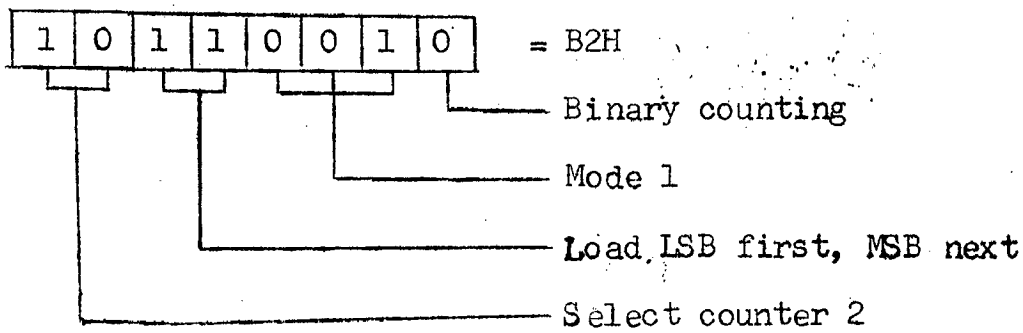


Calling address - F000: 1600.

2) SUBROUTINE SCR

This routine issues firing pulses to the SCR converter bridge. The o/p of zero crossing detector as discussed in SCR firing circuit, retriggers counter 2 of 8253, initialized in mode 1 and loaded with a delay count corresponding to α(predecided). Thus, the firing pulses are issued at an angle α from the 0^{o}.

The mode word taken for initializing counter 2 of 8253 in mode 1 is taken as -



Calling address - F000: 1640

3) SUBROUTINE ADC :

This routine initiates ADC by placing address of the channel to be used on the address lines and sending start of conversion pulse and ALE signal to the ADC through PA of 8255-III. The End of conversion signal is polled through PC_2-PC_0. When it goes high to low and then low to high, the analog to digital conversion is over. The digital equivalent is then inputted through PB of the 8255-III.

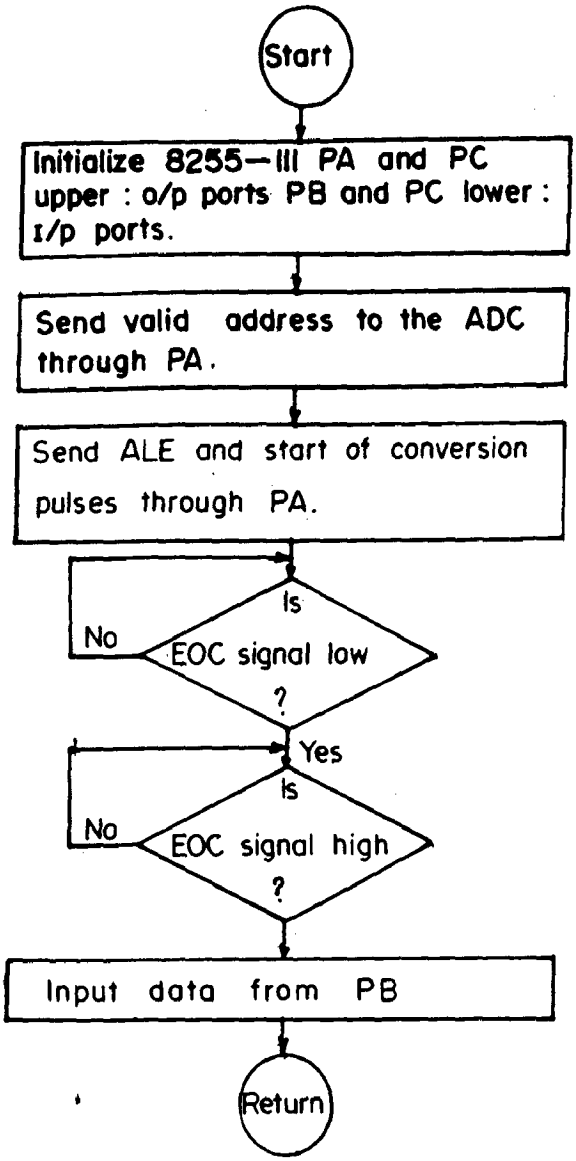
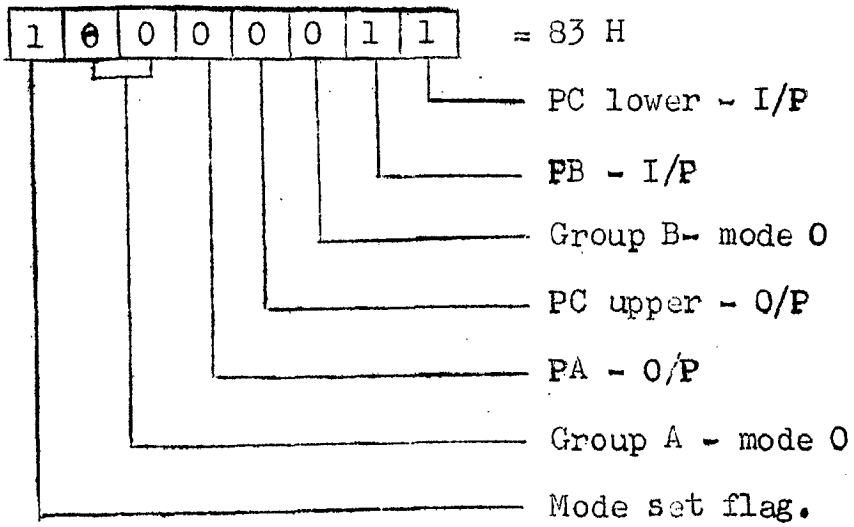


FIG. 6.4.1 : FLOW CHART FOR SUBROUTINE ADC.

8255-III is initialized with PA and PC upper as O/P ports and PC and PC lower as I/P ports. The mode word for initializing 8255-III is taken as



The table 6.4.1 shows the addresses for the 8 channels of each ADC.

TABLE 6.4.1

ADC NO.	CHANNEL NO.	ADDRESSES
1	0-7	00-07
2	0-7	08-0F
3	0-7	10-17

Calling address - F000: 1660.

4) SUBROUTINE FREQ

This routine counts the no. of pulses in one sec.,i.e. the frequency of the pulses. Th pulses are given at PC_0 of 8255-II and polled through it. When a pulse goes high to low

and then low to high, the counter counting no of pulses is incremented by one. This counting is done for 1 sec.

The 1 sec duration is realized by interrupting the μp twenty times, each interrupt at 50 ms. For this counter 0 of 8253 is loaded with a count of 61300_{10} in mode 0. This interrupts the processor through IR4 of 8259. In the I.S.S., it is checked if the processor is interrupted twenty times. If not, counter 0 is reloaded with the same count. The duration achieved by loading counter 0 with a count 61300_{10} twenty times is -

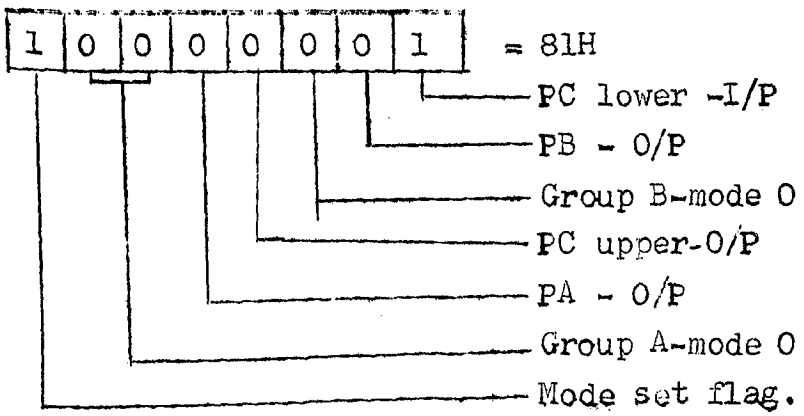
$$\frac{61300}{1.225} \times 20\ \mu s = 1000.81 \times 10^3\ \mu s = 1000.81\ ms,$$

as the clock freq. to timer 0 is 1.225, as discussed in SCR firing circuit.

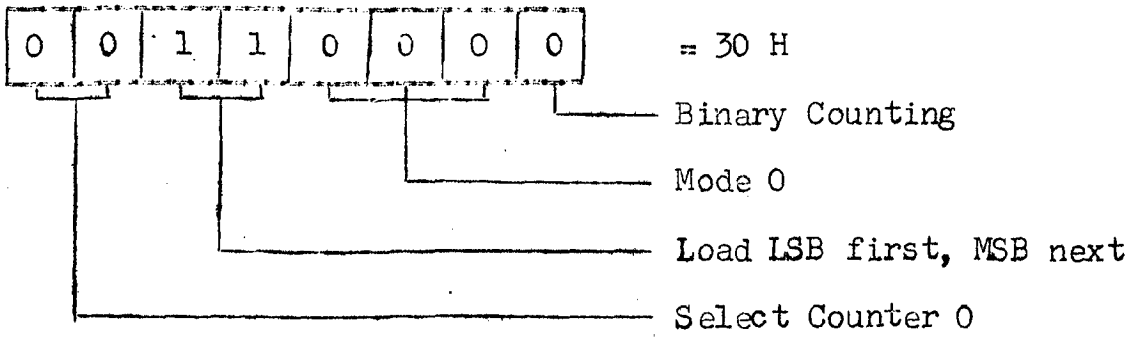
In the total program it is assumed that execution of instructions other than those used for counting the no. of pulses in the 20 interrupt cycles take 0.81 ms. Hence, the pulses are counted effectively for (1000.81-0.81) ms, i.e. 1 sec.

The various mode words for initializing 8255, 8253 and 8259 for this routine are given below.

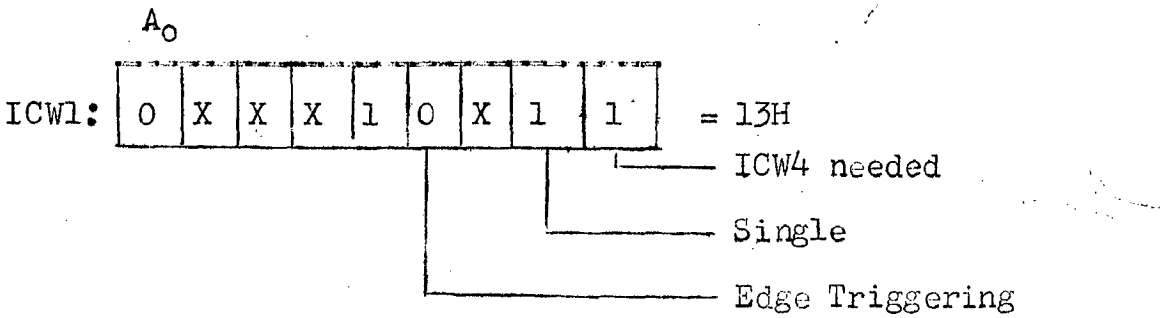
8255 Initialization: PC lower as I/P port and rest all as O/P

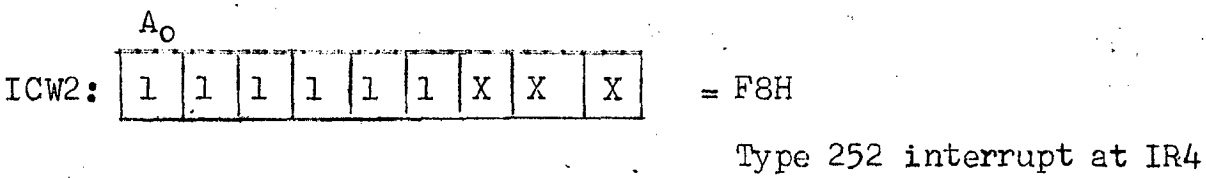


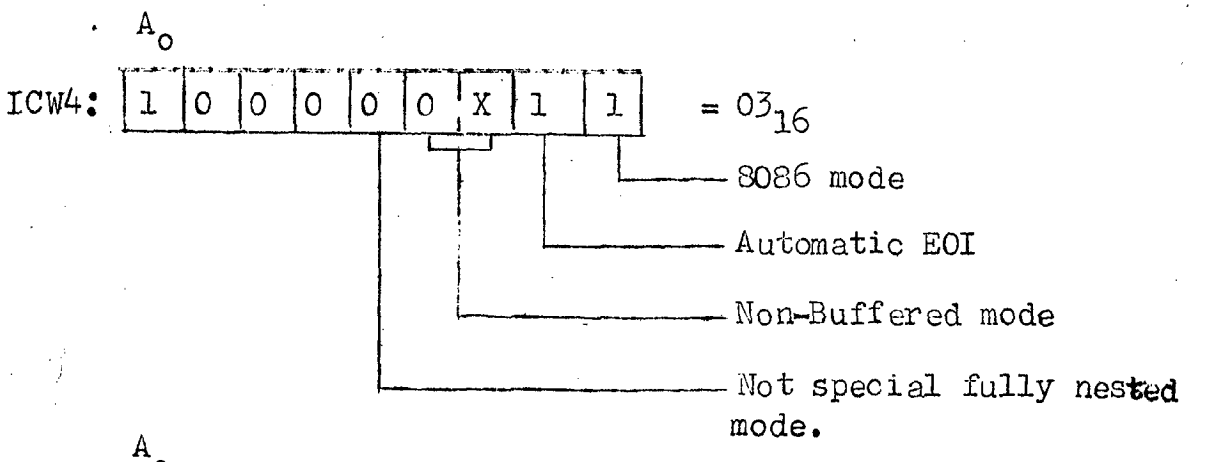
8253 Initialization : Counter 0 in mode 0

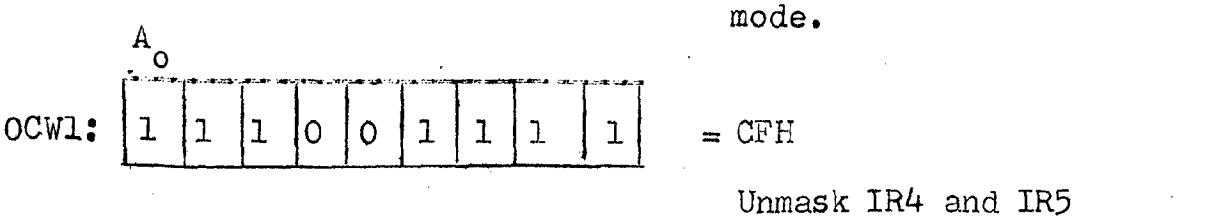


8259 Initialization :









However, for the present work, this routine is not used. This could be used for level control.
Calling address - F000:1690

6.5 TEMPERATURE MONITORING AND OTHER ROUTINES

These routines include PID control routine, its implementation to monitor temperature, necessary data transfer from EPROM to RAM area and conversion of ref. temp. value to equivalent digital form. These routines with flow charts, wherever necessary are described below-

1) SUBROUTINE STAT:

This routine asks for the ref. temp. in BCD through subroutine SETD, converts it into equivalent digital form through subroutine UPBBC and then jumps to subroutine STAT1 to monitor the temperature.

Calling address - F000: 14B0

2) SUBROUTINE STAT1

This routine monitors temp. through PID control scheme. The sampling period is taken as 8 sec. For this, the counter 1 of 8253 is initialized in mode 3, and is loaded with $EF74_{16}$ to generate square pulses of time period 50 ms. The OUT1 is, connected to IR5 of 8259. For a sampling period of 8 sec., the up is to be interrupted 160 times by OUT1 (8259 being initialized to detect edge triggerred interrupts).

At each sampling period, the error is calculated in digital form and stored in locations 0000:3EE0-3EE1. Through PID control scheme, an apparent firing angle is calculated. The apparent firing angle is 160°-actual firing angle (as for a firing angle of 160°, the power delivered by SCR converter bridge is zero, as is shown in Fig. 7.1.1.

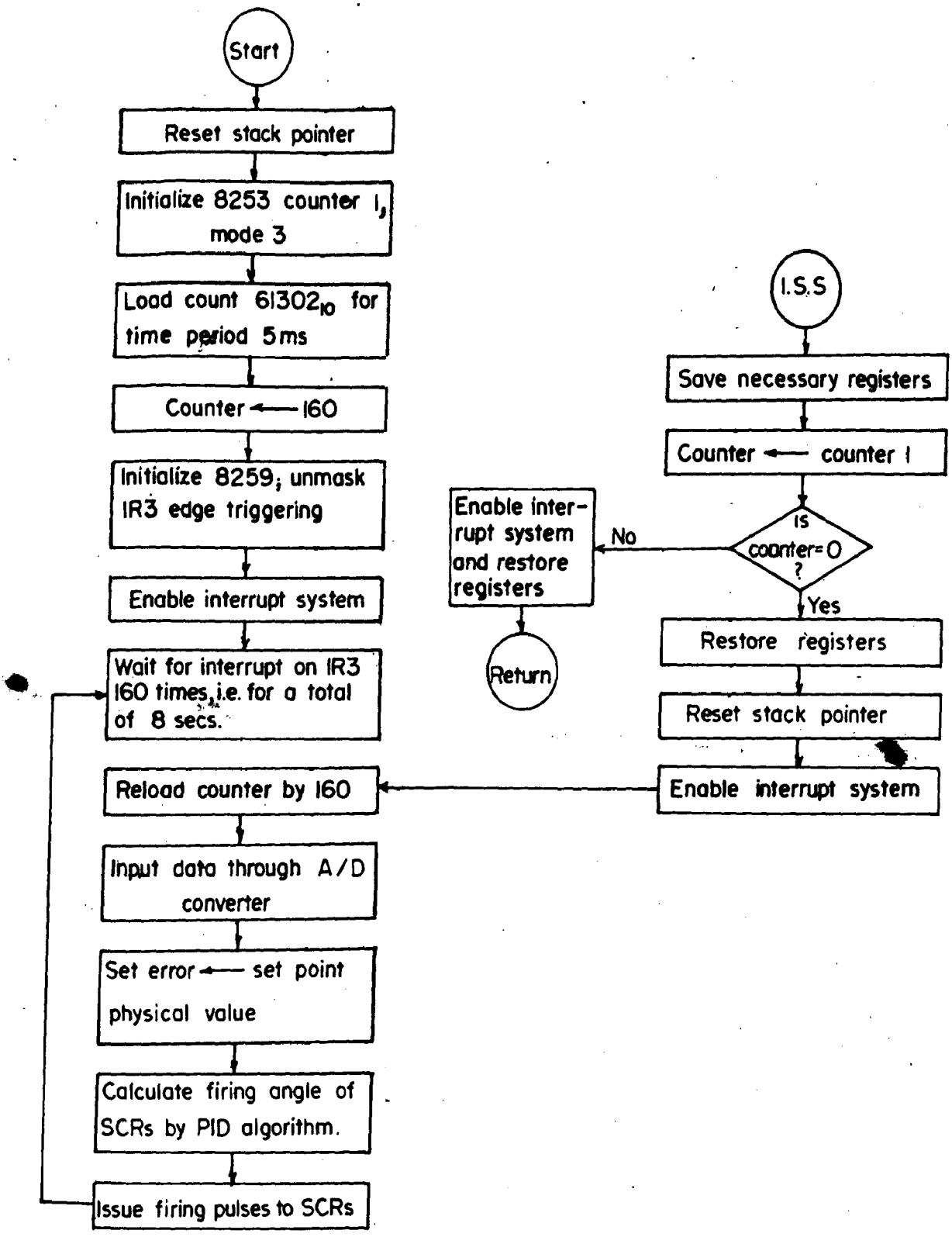
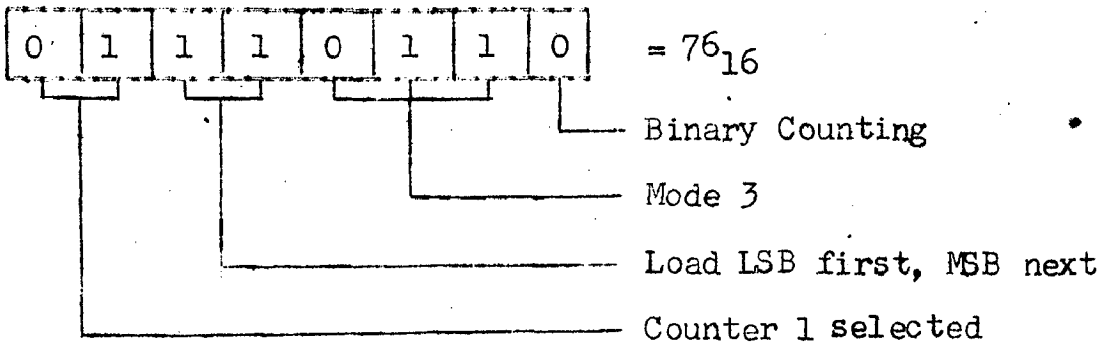


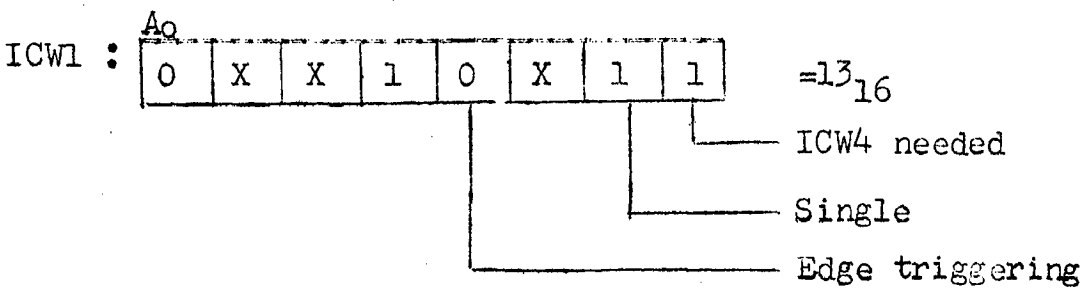
FIG. 6.5.1 : FLOW CHART FOR SUBROUTINE STAT1

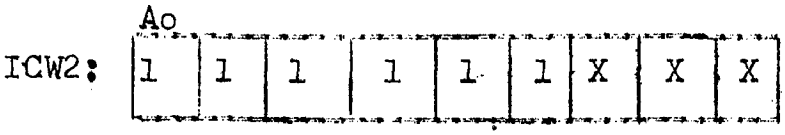
The apparent firing angle (which is in 4 bytes) calculated by PID control scheme is then limited between 0^{o} to 160^{o}. A count in counter 2 is loaded which is directly proportional to the actual firing angle (160-apparent firing angle) and is 68 times of it. If the actual firing angle is 0^{o}, the count to be loaded is 0001. The firing pulses are sent at this calculated firing angle for the current sampling period.

The mode word for initializing counter 1 of 8253 in mode 3 is taken as

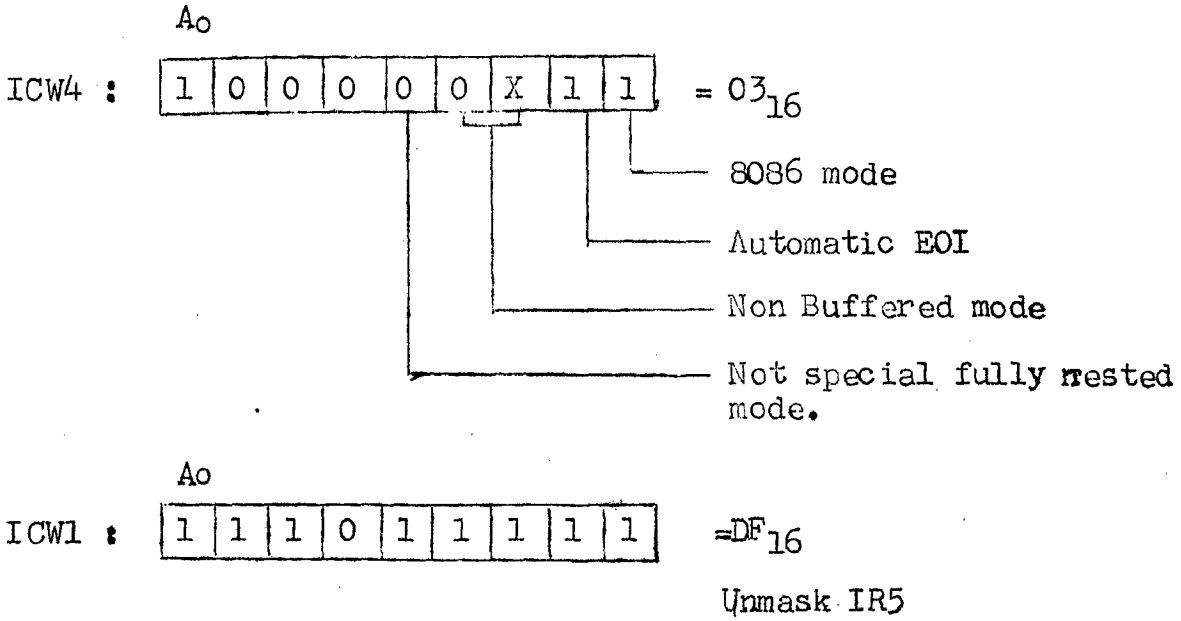


For initializing 8259,





Type 253 interrupt at IR5



The flow chart for this routine is given in Fig. 6.5.1

Calling address - F000: 1700.

3) SUBROUTINE PID :

This routine performs all the PID calculations. For integral term, as discussed earlier, 15 errors(including the current error) are taken. The errors are stored in locations 0000:3F00-3F1D (each error occupying 2 bytes) in the order e_{n-14}- e_n respectively. Values of K, K_I T_S and K_D/T_S are stored in 0000: 3F1E-3F23 (2 bytes for each) respectively. The value of V_m, taken in 4 bytes is stored in 0000: 3F24-3F27.

During intermediate stages of calculations, the integral term is stored in 0000:3F28-3F2B and the derivative term in 0000: 3F2C-3F2F.

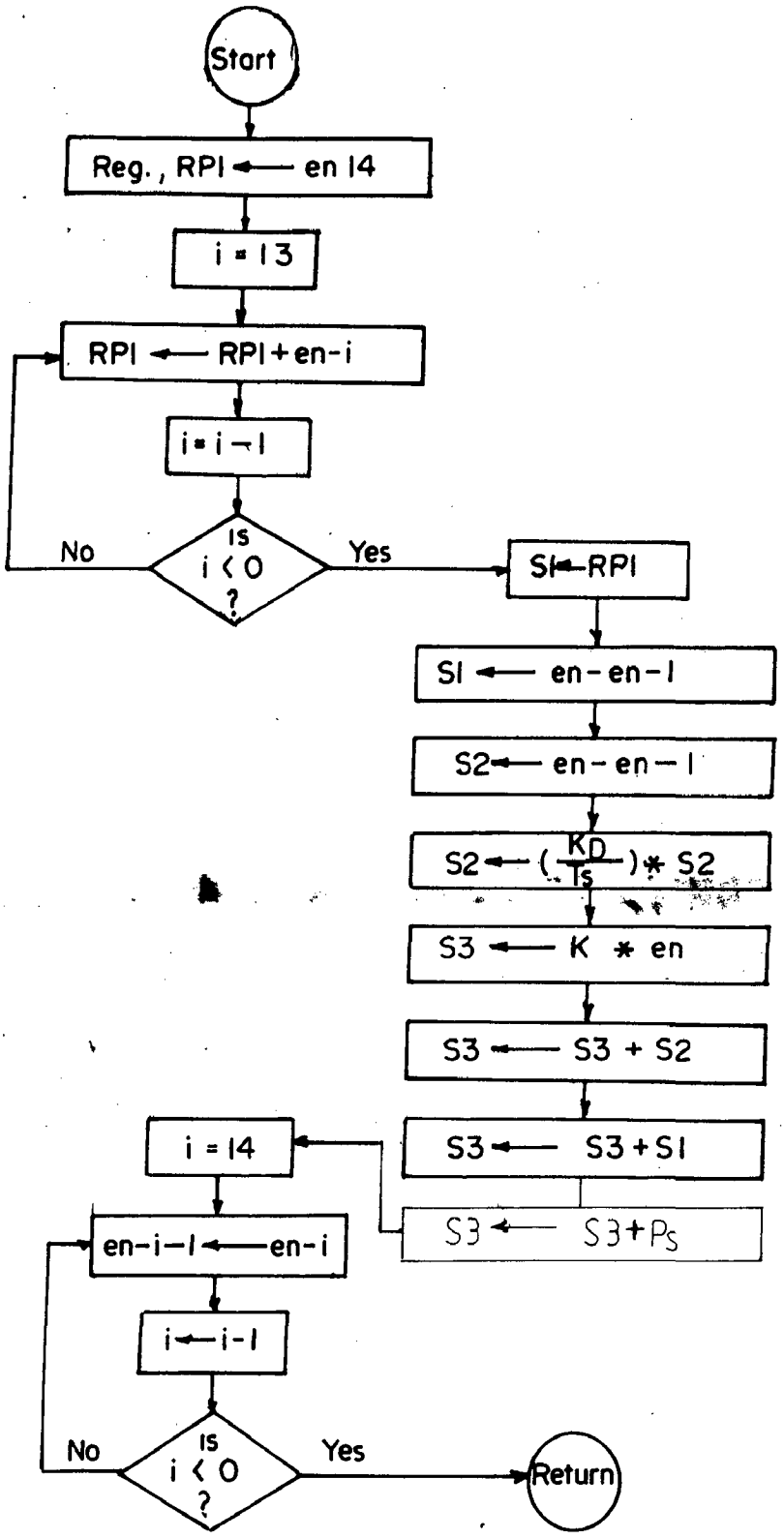


FIG. 6.5.2: FLOW CHART FOR SUBROUTINE PID.

Finally, after all the calculations, $e_i (i=n-13, n)$ are shifted to e_{i-1} positions for processing in the next sampling period

The flow chart for this routine is given in Fig. 6.5.2.

Calling address - F000 : 1780.

4) SUBROUTINE SAPPD:

This routine is used as an Interrupt Service Routine for IR5 to indicate the begining of a sampling pd. of 8 sec. This routine is executed after every 50 ms. When it is executed 160 times, it indicates that 50 x 160 ms=8 sec. sampling pd is over and the new sampling pd. is to start and then it returns to monitor temperature in subroutine STAT1 in the next sampling pd.

Calling address - F000: 17E0.

5) SUBROUTINE UPBBC

This routine converts the ref. temp value stored in locations 0000: 3EF0 - 3EF1 in unpacked BCD format into a digital equivalent that would have been obtained as an o/p of the ADC. This equivalent value in binary is then stored in locations 0000: 3EE0-3EE1.

First, the unpacked BCD value is converted into binary. The thermocouple gives an o/p of 40 uv/^{o}c approximately. This is amplified by an amplifier stage, as discussed in temperature transducer and signal conditioning

to 40 mv/^{o}C. So the binary value of ref temp. is multiplied by this value so as to give an equivalent value of thermo-couple o/p voltage in mV. If this value is fed to the ADC as an I/P, the o/p of ADC would be 0.04 x ref. temp x 255/5, i.e. ref. temp. X 204_{10}/ 100_{10}. Hence, the binary value of ref. temp. is multiplied by 204_{10} and then divided by 100_{10} so as to give an equivalent digital value

Calling address - F000: 1800.

6) SUBROUTINE DTRF:

This routine transfers the necessary data from EPROM to RAM area in the begining of main program. This data includes -

i) The offset and segment addresses for interrupt service routines for IR4 of 8259 (to be used by subroutine FREQ) and that for IR5 (to be used by subroutine STAT 1). This data is transferred from locations F000: 3200-3207 to locations 0000: 03F0-03F7.

ii) Zero is filled in all the locations reserved for e_{n-14} to e_n(i.e. in locations 0000: 3F00-3F1D), to be used by subroutine PID initially.

iii) All the control parameters, i.e., the value of K K_I T_S and K_D/T_S are transferred from locations F000: 3208 - 320D to locations 0000: 3F1E-3F23.

Calling address - F000: 1830.

The total program in the EPROM occupies a memory area F000:0000 - 0013, 1000-1200, 1300-1500, 1600-1860 and the data occupies an area F000:3000- 3210 and 3F00-3F03. The RAM area used by the program is 0000:03F0 - 03F7, 3EA0-3F30.

The program listing is given below.

PROGRAM LISTING

FUNCTION NAME : MAIN-PROGRAM

INPUT : Memory Locations 0000 : 3F24-3F27
F000: 3166,3170-3189, 31A0-31A7,
3F00-3F03, 31B0 -31B9

OUTPUT : None

CALLS : INTF, DTRF, SGNON

DESTROYS : All registers; Memory locations 0000:03F0, 03F7, 3EA0-3EBF, 3EE0-3EE1, 3EF0-3EF1, 3F00-3F23, 3F28-3F2F.

DESCRIPTION : The value of Ps is required to be stored in the memory locations 0000: 3F24-3F27 (four-bytes) before executing the program. The description of main program is already given in section 6.1.

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
	F000: 0000	BB 00 13	MOV BX, INTF	
	0003	FF D3	CALL INTF	
	0005	BB 30 18	MOV BX,DTRF	Transfer data from
	0008	FF D3	CALL DTRF	EPROM to RAM
	000A	BB 90 10	MOV BX,SGNON	Display sign-on
	000D	FF D3	CALL SGNON	message
	000F	BB E0 10	MOV BX,FNCOMM	Wait for functional Commands and
	0012	FF E3	JMP FNCOMM	execute if valid.

FUNCTION NAME : INITN

INPUT : NONE

OUTPUT : NONE

CALLS : NONE

DESTROYS : AL, DX

DESCRIPTION : Initializes 8251 for -
Clock freq. = 64 X baud rate
Stop bit = 1
No. parity, Word length = 8
All the routines developed for serial communication with the CRT assume this routine is already called.

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
INITN	F000:1000	BA F2 FF	MOV DX, FFF2	
	1003	BO 2F	MOV AL, 2F	Initialize 8251 as required
	1005	EE	OUT DX,AL	
	1006	BO 27	MOV, AL,27	Receive and transmit enable
	1008	EE	OUT DX, AL	
	1009	C3	RET	

FUNCTION NAME : CHRIN

INPUT : NONE

OUTPUT : AL

CALLS : NONE

DESTROYS : DX

DESCRIPTION : Inputs the ASCII code of a character pressed on the ASCII keyboard, in the reg. AL. It waits till the receiver is ready.

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
CHRIN CHRI	F000:1010	BA F2 FF	MOV DX,FFF2	
	1013	EC	IN AL,DX	} Get console's
	1014	A8 02	TEST AL,02	} status
	1016	74 FB	JZ CHRI	Wait if receiver is not ready
	1018	BA F0 FF	MOV DX,FFF0	Get char.in AL
	101B	EC	IN AL,DX	}
	101C	C3	RET	

FUNCTION NAME : CHROUT

INPUT : AL

OUTPUT : NONE

CALLS : NONE

DESTROYS : AH, DX

DESCRIPTION : Outputs a char. whose ASCII code is in (AL) to the console

It waits till the transmitter is ready.

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
CHROUT	F000:1020	8A E0	MOV AH,AL	
	1023	BA F2 FF	MOV DX,FFF2	
CHRO	1025	EC	IN AL,DX	Check for trans-
	1026	A8 01	TEST AL,01	} mitter to get ready by reading 8251's status
	1028	74 FB	JZ CHRO	
	102A	8A C4	MOV AL,AH	
	102C	BA F0 FF	MOV DX,FFF0	Data transfer to
	102F	EE	OUT DX, AL	} CRT through 8251
	1030	C3	RET	

FUNCTION NAME : ECHO

INPUT : AL

OUTPUT : NONE

CALLS : CHROUT

DESTROYS : DX

DESCRIPTION : Echoes the character whose ASCII Code is in(AL)on the console.
'ESC' is echoed as 'Ø'
'CR' is echoed as 'LF'+'CR'.

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS.
ECHO	F000:1040	53	PUSH BX	Save BX
	1041	BB 20 10	MOV BX,CHROUT	Pointer for CHROUT subroutine
	1044	50	PUSH AX	Save Char
	1045	3C 1B	CMP Al,'ESC'	See if echoing 'ESC'
	1047	75 02	JNZ L1	No, branch
	1049	B0 24	MOV AL,Ø	Yes,Echo as Ø
L1	104B	FF D3	CALL CHROUT	Output char
	104D	3C 0D	CMP AL,'CR'	Check if 'CR'
	104F	75 04	JNZ L2	No, branch
	1051	B0 0A	MOV AL, 'LF'	Yes, output 'LF'
	1053	FF D3	CALL CHROUT	Indirect call through BX
L2	1055	58	POP AX	Restore character.
	1056	5B	POP BX	Restore BX
	1057	C3	RET.	

FUNCTION NAME : NTIMES

INPUT : AL,CL

OUTPUT : NONE

CALLS : CHROUT

DESTROYS : AH, BX,CL,DX

DESCRIPTION : Sends the character whose ASCII Code is in reg. 'AL' to the console as many times as the contents of reg. 'CL'. If (CL)=0, the character is outputted 256 times.

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
NTIMES	F000:1060	BB 20 10	MOV BX,CHROUT	
L1	1063	FF D3	CALL CHROUT	Send char.to console
	1065	FE C9	DEC CL	till (CL)is decremented
	1067	75 FA	JNZ L1	to zero
	1069	C3	RET	

FUNCTION NAME : STNG

INPUT : CL,SI, DS

OUTPUT : NONE

CALLS : ECHO

DESTROYS : CL,DX,SI

DESCRIPTION : Sends an string of characters whose ASCII Codes are stored in memory locations starting from (DS):(SI). The no. of characters in the string are given by (CL).

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
STNG	F000:1070	57	PUSH DI	Save DI
	1071	FC	CLD	Automatic Increment
	1072	BF 40 10	MOV DI,ECHO	
L1	1075	AC	LODSB	Load byte from (DS):(SI) and auto increment to (SI)
	1076	FF D7	CALL ECHO	
	1078	FE C9	DEC CL	All characters echoed ?
	107A	75 F9	JNZ L1	No, send next char.
	107C	5F	POP DI	Restore DI if yes
	107D	C3	RET	

FUNCTION NAME	: ERROR
INPUT	: NONE
OUTPUT	: NONE
CALLS	: CHROUT
DESTROYS	: AX,BX,DX
DESCRIPTION	: Sends 'M' as an error message to the console.

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
ERROR	F000:1080	BB 20 10	MOV BX,CHROUT	
	1083	B0 2A	MOV Al,'M'	
	1085	FF D3	CALL CHROUT	Send 'M' to console.
	1087	C3	RET	

FUNCTION NAME : SGNON

INPUT : Mem. Locations F000:3000-3157

OUTPUT : None

CALLS : ECHO,NTIMES,STNG

DESTROYS : AX, BX, CL,DX,SI,DI,BP,DS,ES.

This routine clears the whole CRT screen and displays the sign-on message.

DESCRIPTION : The ASCII codes for each character are stored in the memory locations F000:3000-3157 The starting location is named as START.

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
SGNON	F000:1090	BD 16 01	MOV BP,NCHAR	No.of mem.locations in sign-on-table
	1093	FC	CLD	
	1094	BB 00 F0	MOV BX,Data Seg.	
	1097	8E DB	MOV DS,BX	
	1099	BB 40 10	MOV BX,ECHO	
	109C	BE 00 30	MOV SI,START	Starting address of the table.
	109F	BF 60 10	MOV DI,NTIMES	
	10A2	B1 20	MOV CL,20	20_{16}line feeds to
	10A4	B0 0A	MOV A1,'LF' }	blank the console
	10A6	FF D7	CALL NTIMES	
	10A8	AC	LODSB	
	10A9	3C FF	CMP AL,FF	Is data FF ?

contd...

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
	10AB	75 0A	JNZ L1	No,branch
	10AD	B0 20	MOV AL, ' '	Yes 'Space' in (AL)
	10AF	8A 0C	MOV CL,(SI)	No.of spaces in (CL)
	10B1	46	INC SI	
	10B2	4D	DEC BP	
	10B3	FE C9	DEC CL	One space less
	10B5	FF D7	CALL NTIMES	Send spaces to console
L1	10B7	BB 40 10	MOV BX, ECHO	Echo the char.
	10BA	FF D3	CALL ECHO	
	10BC	4D	DEC BP	All char.sent ?
	10BD	75 E9	JNZ L2	No. send next.
	10BF	B1 02	MOV CL,NFL	No. of 'LF' to be given after SGNON message.
	10C1	B0 0A	MOV AL,'LF'	
	10C3	FF D7	CALL NTIMES	
	10C5	BB 70 10	MOV BX,STNG	Send Guided by message
	10C8	B1 42	MOV CL,NGD	No.of char in Guided by message
	10CA	FF D3	CALL STNG	
	10CC	C3	RET	

ASCII CODES FOR SGNON MESSAGE:

```
START DW 3000

ASSUME DS: F000

F000:3000 .DB  FF 08 23 20 23 20 23 FF 05 23 20 23 20 23 FF 04
     3010      23 20 23 20 23 20 23 20 23 FF 03 23 20 23 20 23
     3020      20 23 0D FF 06 23 FF 0A 23 FF 0D 23 FF 07 23 FF
     3030      06 23 0D FF 06 23 FF 0B 23 20 23 20 23 FF 08 23
     3040      FF 07 23 20 23 20 23 20 23 0D FF 06 23 FF 10 23
     3050      FF 07 23 FF 07 23 FF 05 23 0D FF 08 23 20 23 20
     3060      23 FF 05 23 20 23 20 23 FF 08 23 FF 07 23 FF 06
     3070      23 0D 0A 0A 0A 0A FF 11 23 20 23 20 23 20 23 FF
     3080      04 23 20 23 20 23 20 23 FF 07 23 FF 07 23 20 23
     3090      20 23 20 23 FF 04 23 FF 05 23 0D FF 11 23 FF 06
     30A0      23 FF 03 23 FF 0C 23 20 23 FF 06 23 FF 06 23 FF
     30B0      04 23 FF 03 23 0D FF 11 23 20 23 20 23 20 23 FF
     30C0      04 23 20 23 20 23 FF 07 23 FF 03 23 FF 05 23 FF
     30D0      06 23 FF 05 23 20 23 0D FF 11 23 FF 05 23 FF 04
     30E0      23 FF 0A 23 20 23 20 23 20 23 FF 04 23 FF 06 23
     30F0      FF 06 23 0D FF 11 23 FF 06 23 FF 03 23 20 23 20
     3100      23 20 23 FF 03 23 FF 07 23 FF 03 23 20 23 20 23
     3110      20 23 FF 07 23 0D 20 20 20 20 20 20 47 55 49 44
     3120      45 44 20 42 59 3A 31 29 53 48 2E 4D 2E 4B 2E 56
     3130      41 53 41 4E 54 48 41 0D 20 20 20 20 20 20 20 20
     3140      20 20 20 20 20 20 20 20 32 29 53 48 2E 42 2E 4D
     3150      4F 48 41 4E 54 59 0D 0D
```

FUNCTION NAME : FNCOMM

INPUT :
MEM. LOCATIONS F000:3170-3189
F000:31A0-31A7
F000:3F00-3F03

OUTPUT : NONE

CALLS : ECHO,CHROUT,CHRIN,HELP,STAT

DESTROYS : AX,BX,CX,DX,SI,DI,SP,DS,ES,SS
Mem. locations 0000:3EA0-3EBF

DESCRIPTION : Waits for a functional command to be given by displaying a prompt character. The ASCII Codes of the given command are stored in locations 0000: 3EA0-3EBF. It compares these codes with the ASCII codes of valid commands stored in locations F000: 31A0-31A7. If the command is valid, the address for executing that command is taken from mem. locations F000:3F00-3F03. For invalid commands a message, 'HELP' FOR VALID COMMANDS is displayed, the ASCII Codes of which are stored in F000: 3170-3189.

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
FNCOMM	F000:10E0	B9 00 00	MOV CX,0000	} Set stack segment to
	10E3	8E D1	MOV SS,CX	(SS) ⟵ 0000
	10E5	BC 00 01	MOV SP,0100	(SP) ⟵ 0100
L4	10E8	BB 40 10	MOV BX, ECHO	
	10EB	B0 0D	MOV AL,'CR'	} Send 'CR' and
	10ED	FF D3	CALL ECHO	'LF' to console
	10EF	BB 20 10	MOV BX,CHROUT	

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
	10F2	B0 2E	MOV AL, '.'	Send prompt
	10F4	FF D3	CALL CHROUT	char
	10F6	FC	CLD	Clear direction flag
	10F7	8E C1	MOV ES,CX	(ES) ←0000
	10F9	BF A0 3E	MOV DI,BUFFER	To store commands char.in buffer.
	10FC	Bl 00	MOV CL, 00	Counter for no. of char.in the command given.
L3	10FE	BB 10 10	MOV BX,CHRIN	Input a character
	1101	FF D3	CALL CHRIN	
	1103	BB 40 10	MOV BX,ECHO	Echo it
	1106	FF D3	CALL ECHO	
	1108	3C 0D	CMP AL, 'CR'	Compare if 'CR'
	110A	74 1F	JZ L1	
	110C	3C 2F	CMP AL, '/'	Is it a delete char.
	110E	74 05	JZ L2	Yes,branch
	1110	AA	STDS B	Store in buffer.
	1111	FE C1	INC CL	Increment Counter for no. of chars. given
	1113	EB E9	JMP L3	
L2	1115	4F	DEC DI	Point to previous char.in buffer to delete it
	1116	FE C9	DEC CL	No.of chars. to be decremented by 1
	1118	79 E4	JNS L3	

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
ERR	111A	BB 80 10	MOV BX,ERROR	Error if rubbing
	111D	FF D3	CALL ERROR	(deleting)unnecessarily
	112F	BB 70 10	MOV BX,STNG	Give the
	1122	BE 70 31	MOV SI,HELMES	message -
	1125	B1 1A	MOV CL,NCHAR	'HELP' FOR
	1127	FF D3	CALL STNG	VALID COMMANDS
	112A	EB BD	JMP L4	Search for next command.
L1	112B	80 F9 04	CMP CL,04	Check if no. of char.
	112E	75 EA	JNZ ERR	is 4.if not, error.
				If yes, compare buffer with fn al table FNCTL
	1130	B0 02	MOV AL,NFC	No. of fn.al commands
	1132	8A C8	MOV CL,AL	Save in CL
	1134	D0 E0	SHL AL,01	Multiply by 4.
	1136	D0 E0	SHL AL,01	to find total no.of char.in table.
	1138	FE C8	DEC AL	SI will now
	113A	B4 00	MOV AH,00	point to end of
	113C	BE A0 31	MOV SI,FNCTL	FNCTL TABLE
	113E	03 F0	ADD SI, AX	
	1141	88 C8	MOV AL,CL	
	1143	FD	STD	Auto decrement.

LEBEL	ADDRESS	COMMENTS	MNEMONICS AND OPERANDS	COMMENTS
L6	1144	BF A3 3E	MOV DI,BUFFER+03	Point to last.char. in buffer.
	1147	B9 04 00	MOV CX,0004	
	114A	F3 A6	REPE CMPS B	CX is decremented by no. of characters which are same.
	114C	74 08	JZ L5	
	114E	FE C8	DEC AL	Compared all commands.
	1150	74 C8	JZ ERR	Yes, Error.
	1152	2B F1	SUB SI, CX	No, decrease SI by CX to point to end of prev. command.
	1154	EB EE	JMP L6	
L5	1156	B4 3F	MOV AH,High byte	Higher byte of address from where address for fn.al command service routine is to be taken
	1158	FE C8	DEC AL	
	115A	D0 E0	SHL Al,01	AL=00 for Ist comm. =02 for 2nd comm. and so on.
	115C	8B D8	MOV BX,AX	
	115E	FF 17	CALL BX	Indirect call within segment.
	1160	BB E0 10	MOV BX,FNCOMM	Jump to serve
	1163	FF E3	JMP FNCOMM	Fn.al Comm.

ASCII CODES FOR 'HELP' FOR VALID COMMANDS MESSAGE:

```
HELMES DW   3170

ASSUME DS: F000

F000:3170  DB  27 48 45 4C 50 27 20 46 4F 52 20 56 41 4C 49 44
     3180      20 43 4F 4D 4D 41 4E 44 53 0D
```

ASCII CODES FOR THE VALID COMMANDS:

```
FNCTL  DW  31A0

ASSUME DS; F000

F000:31A0 DB  48 45 4C 50 53 54 41 54
```

ADDRESS TO SERVE VALID COMMANDS

```
ASSUME DS : F000

F000:3F00 DB 80 11 B0 14
```

FUNCTION NAME : HELP

INPUT : LOCATIONS F000:31A0 -31A7

OUTPUT : NONE

CALLS : ECHO,STNG

DESTROYS : AX,BX,CX,DX,SI

DESCRIPTION : This routine displays the list of valid commands as -

1) HELP
2) STAT

The ASCII codes for these are stored in locations F000: 31A0-31A7.

LEBEL	ADDRESS	CONTENTS	MENMONICS AND OPERANDS	COMMENTS
HELP	F000:1180	B4 02	MOV AH,02	No.of comm.to be displayed
	1182	46	INC SI	Point to begining of FNCTL table
	1183	B5 31	MOV CH,31	ASCII code for 1
L1	1185	BB 40 10	MOV BX,ECHO	
	1188	B0 0D	MOV AL,'CR'	
	118A	FF D3	CALL ECHO	Echo 'CR'
	118C	8A C5	MOV AL,CH	ASCII code for comm. no.
	118E	FF D3	CALL ECHO	Send comm.no.to console
	1190	B0 29	MOV AL, ')'	
	1192	FF D3	CALL ECHO	Send')' to console
	1194	B0 20	MOV AL,'$'	
	1196	FF D3	CALL ECHO	Send '$' to console

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
	1198	B1 04	MOV CL,04	No. of char.in a command.
	119A	BB 70 10	MOV BX,STNG	
	119D	FF D3	CALL STNG	Send comm.pointed by (SI)
	119F	FE C5	INC CH	
	11A1	FE CC	DEC AH	All commands sent ?
	11A3	75 E0	JNZ L1	No, branch.
	11A5	C3	RET	

FUNCTION NAME : VALID

INPUT : AL

OUTPUT : NONE

CALLS : ERROR

DESTROYS : AL, BX,DX

DESCRIPTION : Checks if a char. in AL is a BCD digit or not. If not, stores '*' in AL and echos it.

Also, for valid char., it convets ASCII char.into unpacked BCD format.

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
VALID	F000:11B0	2C 30	SUB AL,30	Convert to unpacked BCD
	11B2	78 04	JS ERR	Error, if ASCII code less than 30.
	11B4	3C 09	CMP AL,09	Error if BCD Char.>9
	11B6	7E 05	JLE L1	
ERR	11B8	BB 80 10	MOV BX,ERROR	Echos '*' and stores in AL.
	11BB	FF D3	CALL ERROR	
L1	11BD	C3	RET	

FUNCTION NAME : SETP

INPUT : Mem.locations F000:31B0-31B9

OUTPUT : Mem. Locations0000:3EF0-3EF1

CALLS : ECHO,STNG,CHRIN,VALID

DESTROYS : AX,BX,CX,DX,SI,DI
Mem. locations 0000:3EF0-3EF1

DESCRIPTION : Displays 'REF.TEMP:' (ASCII codes stored in F000:31B0-31B8), and waits for the BCD set point to be given. The set point in unpacked. BCD format is stored in RAM area 0000: 3EF0-3EF1. If nothing is given, the ref. temp. will be taken as 40. This routine is already described before.

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
STEP	F000:11C0	FC	CLD	Auto increment
	11C1	BF F0 3E	MOV DI,SETPT	Point to begining of RAM area where ref. temp is to be stored.
	11C4	BE B0 31	MOV SI, PAR	Point to table where ASCII, codes of 'REF. TEMP:are stored.
	11C7	BB 70 10	MOV BX,STNG	
	11CA	B1 0A	MOV CL,0A	Display 'REF.TEMP:'
	11CC	FF D3	CALL STNG	
	11CE	B9 00 04	MOV CX, 0400	Move Ref.Temp←40 °C.
	11D1	90	NOP	
	11D2	90	NOP	
	11D3	90	NOP	

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
	11D4	90	NOP	
	11D5	90	NOP	
L2	11D6	BB 10 10	MOV BX,CHRIN	
	11D9	FF D3	CALL CHRIN	Input character
	11DB	BB 40 10	MOV BX,ECHO	
	11DE	FF D3	CALL ECHO	Echo it
	11E0	3C 0D	CMP AL,'CR'	Is it 'CR'?
	11E2	74 0F	JZ L1	Yes,branch
	11E4	BB B0 11	MOV BX,VALID	
	11E7	FF D3	CALL VALID	Display '*' if invalid char.
	11E9	3C 2A	CMP AL, '*'	Is char.invalid ?
	11EB	74 E9	JZ L2	Yes,branch
	11ED	8A E9	MOV CH,CL	(CX) ←Higher digit.
	11EF	8A C8	MOV CL,AL	(CL) ← Lower digit
	11F1	EB E3	JMP L2	
L1	11F3	89 C8	MOV AX,CX	(AX) ←Unpacked BCD value.
	11F5	AB	STDS W	
	11F6	C3	RET	

ASCII CODES FOR 'CR REF.TEMP'.

```
PAR         DW   31B0
ASSUME      DS:  F000
F000: 3180 DB 0D 52 45 46 2E 54 45 4D 50 3A
```

FUNCTION NAME : INTF

INPUT : AX

OUTPUT : AX,BL

CALLS : NONE

DESTROYS : AX,BX

DESCRIPTION : An integer in (AX) is returned as a normalised floating point no. with mantissa in (AX) and exponent in (BL).

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
INTF	F000:1300	B7 00	MOV BH,00	Flag to indicate -ve no.
	1302	B3 4F	MOV BL,4F	Set exponent ← 4F
	1304	A9 FF 7F	TEST AX,7FFF	Is no. zero ?
	1307	74 13	JZ L3	Yes branch
	1309	F6 C4 80	TEST AH,80	Check if no.is-ve
	130C	79 04	JNS L1	
	130E	FE C7	INC BH	If negative,
	1310	F7 D8	NEG AX	make +ve
L1	1312	F6 C4 40	TEST AH,40	Check if normalised
	1315	75 21	JNZ STSGN	If yes,set sign
	1317	D1 E0	SHL AX,01	Decrease exp.by
	1319	4B	DEC BL	1 and left shift.

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
	131A	EB F6	JMP L1	
L3	131C	F6 C4 80	TEST AH,80	
	131F	74 0C	JZ UFLOWE	
	1321	B4 C0	MOV AH,C0	
	1323	FE C3	INC BL	
	1325	C3	RET	
UFLOW	132A	B8 00 00	MOV AX,0000	Mantissa underflow
UFLOWE	132D	B3 00	MOV BL,00	Exponent underflow
	132F	C3	RET	
STSGN	1338	F6 C7 01	TEST BH,01	Is no. - ve ie. LSB of BH=1?
	133B	74 02	JZ L5	No,branch
	133D	F7 D8	NEG AX	Yes,Negate no.
L5	133F	C3	RET	

FUNCTION NAME : FTIN

INPUT : AX,BL

OUTPUT : AX

CALLS : NONE

DESTROYS : AX, BX,CL

DESCRIPTION : Converts a floating point no. in (AX)(BL) into an integer in (AX). Overflow is set when result exceeds. 16 bits and an underflow is set when result is less than 0001.

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
FTIN	F000:1350	B7 4F	MOV BH,4F	
	1352	2A FB	SUB BH,BL	Regd.amount of right shift
	1354	78 14	JS OVRFLM	Is exponent > 4F ?
	1356	80 FF 0F	CMP BH,0F	No, is shift $>15_{10}$?
	1359	7F CF	JG UFLOW	Yes,underflow
	135B	8A CF	MOV CL,BH	No,shift arithmetic
	135D	D3 F8	SAR AX,CL	} right
	135F	C3	RET	Return

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
OVRFLOW	F000:1368	B3 7F	MOV BL,7F	Set exponent to 7F
OVRFLM	136A	08 E4	OR AH, AH	Is (AX) - ve ?
	136C	78 04	JS NEGo	Yes, branch
	136E	B8 FF 7F	MOV AX,7FFF	No, +ve overflow
	1371	C3	RET	
NEGO	1372	B8 01 80	MOV AX,8001	-ve overflow
	1375	C3	RET	

FUNCTION NAME : NORMA

INPUT : AX,BL

OUTPUT : AX,BL

CALLS : UFLOW,OVRFLOW,STSGN

DESTROYS : AX,BX

DESCRIPTION : A floating point no. in (AX) (BL) is returned with a normalised floating point no. in (AX) (BL)

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
NORMA	F000:1380	52	PUSH DX	Save DX
	1381	30 FF	XOR BH,BH	Clear BH to check for -ve no.
	1383	A9 FF 7F	TEST AX,7FFF	
	1386	75 1B	JNZ NORM1	
	1388	F6 C4 80	TEST AH,80	Is mantissa 8000 ?
	138B	74 08	JZ L2	No,branch to underflow as mantissa is 0.
	138D	B4 C0	MOV AH,C0	Yes,set mantissa C000
	138F	FE C3	INC BL	Increase exponent by 1
	1391	78 09	JS L3	If overflow,branch
	1393	EB 27	JMP RETURN	Return
L2	1395	BA 2A 13	MOV DX,UFLOW	Set underflow
	1398	FF D2	CALL UFLOW	
	139A	EB 20	JMP RETURN	

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
L3	139C	BA 68 13	MOV DX,OVRFLOW	Set overflow
	139F	FF D2	CALL OVRFLOW	
	13A1	EB 19	JMP RETURN	
NORM1	13A3	09 C0	OR AX,AX	Is no - ve ?
	13A5	79 04	JNS L1	No, branch
	13A7	FE C7	INC BH	Yes, Set flag for -ve no.
	13A9	F7 D8	NEG AX	Negate mantissa
L1	13AB	F6 C4 40	TEST AH,40	Is no normalised ?
	13AE	75 07	JNZ L4	Yes, branch
	13B0	D1 E0	SHL AX,01	No, shift mantissa left.
	13B2	4B	DEC BL	Decrease exp by 1
	13B3	78 E0	JS L2	Underflow if exp.<0
	13B5	EB F4	JMP SHORT L1	Check again
L4	13B7	BA 38 13	MOV DX,STSGN	Set sign of the normalised no.
	13BA	FF D2	CALL STSGN	
RETURN	13BC	5A	POP DX	Restore DX
	13BD	C3	RET	Return.

FUNCTION NAME : SUBTRN/ADDN

INPUT : AX,BX,DX

OUTPUT : AX,BL

CALLS : NORMA,OVERFLOW

DESTROYS : AX,BX,CL,DX

DESCRIPTION : The routine ADDN adds the two normalised operands in (DX)(BH) and (AX)(BL). The routine SUBTRN subtracts the later from the former. The result of both the operations is returned in (AX)(BL). Thus, the routine has two entries one for addition and another for subtraction.

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
SUBTRN	F000:13D0	F7 D8	NEG AX	
ADDN	13D2	88 D9	MOV CL,BL	
	13D4	28 F9	SUB CL,BH	Exponent difference in CL
	13D6	74 0F	JZ ADDM	If diff=0,branch
	13D8	79 06	JNS SHIFT	If diff.>0,branch
	13DA	F6 D9	NEG CL	(CL) ←Magnitude of diff.
	13DC	8A DF	MOV BL,BH	Larger magnitude no. in (AX)(BL)
	13DE	87 C2	XCHG AX,DX	
SHIFT	13E0	80 F9 0F	CMP CL,0F	Diff.magnitude >0F ?
	13E3	7F 1C	JG RETURN	Yes,return
	13E5	D3 FA	SAR DX,CL	No.equalise the exponents

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
ADDM	13E7	03 C2	ADD AX,DX	Add mantissas
	13E9	BA 80 13	MOV DX,NORMA	Address of normalization routine
	13EC	70 04	JO ADDO	If Addition overflow, branch
	13EE	FF D2	CALL NORMA	Normalise.
	13F0	EB 0F	JMP RETURN	
ADDO	13F2	D1 D8	RCR AX,01	Shift right through carry
	13F4	FE C3	INC BL	Increase exp by 1
	13F6	FF D2	CALL NORMA	Normalise
	13F8	08 DB	OR BL,BL	Exponent's sign bit set ?
	13FA	79 05	JNS RETURN	No, return
	13FC	BA 68 13	MOV DX,OVRFLOW	Yes, overflow
	13FF	FF D2	CALL OVRFLOW	
RETURN	1401	C3	RET	

FUNCTION NAME : MULT

INPUT : AX,BX,DX

OUTPUT : AX,BL

CALLS : OVRFLOW, UFLOW, NORMA

DESTROYS : AX,BX,CX,DX

DESCRIPTION : This routine multiplies the two normalised operands in (DX)(BH) and (AX)(BL) and returns the normalised result in (AX)(BL).

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
MULT	F000:1410	02 DF	ADD BL,BH	Add exponents
	1412	B7 00	MOV BH, 00	Indicates overflow/ underflow
	1414	79 02	JNS L1	Branch if sign flag set.
	1416	FE C7	INC BH	
L1	1418	F7 EA	IMUL DX	Integer multiply mantissas
	141A	87 C2	XCHG AX,DX	Higher order result in AX
	141C	80 EB 40	SUB BL,40	Subtract bias
	141F	79 14	JNS L4	
	1421	08 FF	OR BH,BH	
	1423	75 02	JNZ L2	
	1425	EB 07	JMP L3	

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
L2	1427	BA 68 13	MOV DX,OVRFLOW	Set overflow
	142A	FF D2	CALL OVRFLOW	
	142C	EB 10	JMP RETURN	Return
L3	142E	BA 2A 13	MOV DX,UFLOW	Set underflow
	1431	FF D2	CALL UFLOW	
	1433	EB 09	JMP RETURN	Return
L4	1435	D0 E6	SHL DH,01	Shift result
	1437	D1 D0	RCL AX,01	} left once
	1439	BA 80 13	MOV DX,NORMA	
	143C	FF D2	CALL NORMA	Normalise
RETURN	143E	C3	RET	Return

FUNCTION NAME DIV

INPUT AX,BX,DX

OUTPUT AX,BL

CALLS OVRFLOW,UFLOW,NORMA,STSGN

DESTROYS AX,BX,CX,DX

DESCRIPTION This routine divides the normalised operand in (DX).(BH) by a normalised operand in (AX) (BL) and returns the result in (AX)(BL).

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
DIV	F000:1450	8B C8	MOV CX,AX	Save denom.in CX
	1452	2A FB	SUB BH,BL	
	1454	8A DF	MOV BL,BH	Exponent diff.in BL
	1456	B7 00	MOV BH,00	
	1458	8A C7	MOV AL,BH	AL indicates underflow/overflow
	145A	79 02	JNS L1	
	145C	FE C0	INC AL	
L1	145E	80 C3 40	ADD BL,40	Add bias
	1461	79 14	JNS L2	
	1463	08 C0	OR AL,AL	
	1465	75 09	JNZ L6	Underflow,whensign before and after adding 40.
	1467	32 E6	XOR AH,DH	MSB decides sign of overflowed result.

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
	1469	BA 68 13	MOV DX,OVRFLOW	
	146C	FF D2	CALL OVRFLOW	Set overflow
	146E	EB 2F	JMP RETURN	
L6	1470	BA 2A 13	MOV DX,UFLOW	
	1473	FF D2	CALL UFLOW	Set underflow
	1475	EB 2A	JMP RETURN	
L2	1477	31 C0	XOR AX,AX	Clear AX
	1479	08 F6	OR DH,DH	Is dividend -ve ?
	147B	79 04	JNS L3	No,branch
	147D	FE C7	INC BH	Negative mantissa flag
	147F	F7 DA	NEG DX	Make +ve
L3	1481	08 ED	OR CH,CH	Is divider-ve ?
	1483	79 04	JNS L4	No,branch
	1485	FE C7	INC BH	
	1487	F7 D9	NEG CX	Make denom.+ve
L4	1489	D1 E1	SHL CX,01	Shift'denom.left once
	148B	F7 F1	DIV CX	Divide
	148D	08 E4	OR AH,AH	Is quotient's MSB Set?
	148F	79 04	JNS L5	No,branch

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
	1491	D1 E8	SHR AX,01	Yes, shift result right
	1493	FE C3	INC BL	Increment exp.by 1
L5	1495	BA 38 13	MOV DX,STSGN	
	1498	FF D2	CALL STSGN	Set sign
	149A	BA 80 13	MOV DX,NORMA	
	149D	FF D2	CALL NORMA	Normalise
RETURN	149F	C3	RET	

FUNCTION NAME STPM

INPUT NONE

OUTPUT NONE

CALLS DELAY

DESTROYS BX,AL,DX

DESCRIPTION This routine rotates the stepper motor through 1.8^{o} by issuing pulses from PA of 8255-I

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
STPM	F000:1600	B0 80	MOV AL,80	Initialize all
	1602	BA FE FF	MOV DX,FFFE	} ports as o/p
	1605	EE	OUT DX,AL	ports
	1606	BA F8 FF	MOV DX,FFF8	o/p code
	1609	B0 FA	MOV AL,FA	} for.
	160B	EE	OUT DX,AL	step 0
	160C	BB 30 16	MOV BX,DELAY	Delay between
	160F	FF D3	CALL DELAY	} 2 steps
	1611	B0 F6	MOV AL,F6	o/p code for
	1613	EE	OUT DX,AL	} step 1
	1614	FF D3	CALL DELAY	Delay between 2 steps
	1616	B0 F5	MOV AL,F5	o/p code for step 3
	1618	EE	OUT DX,AL	
	1619	FF D3	CALL DELAY	

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
	161B	B0 F9	MOV AL,F9	o/p code for step 4.
	161D	EE	OUT DX,AL	
	161E	FF D3	CALL DELAY	
	1620	C3	RET	
DELAY	1630	50	PUSH AX	Save reg.
	1631	B8 00 0F	MOV AX,0F00	Count for 30 ms delay
L1	1634	48	DEC AX	Decrement count
	1635	75 FD	JNZ L1	Is count zero?
	1637	58	POP AX	Yes, restore reg.
	1638	C3	RET	

FUNCTION NAME SCR

INPUT BX

OUTPUT NONE

CALLS NONE

DESTROYS AL,DX

DESCRIPTION This routine issues the firing pulses to SCR converter bridge at an angle α, which corresponds to a delay count in (BX) to be loaded into the timer. The delay is with reference to the rising edge o/p of zero crossing detecter.

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
	F000:1640	BA DE FF	MOV DX,Control word	
	1643	BO B2	MOV AL,B2	Counter 2, LSB first, MSB next, mode 1, Binary counting
	1645	EE	OUT DX,AL	
	1646	BA DC FF	MOV DX,Counter 2	
	1649	8A C3	MOV AL,BL	Load LSB into
	164B	EE	OUT DX,AL	Counter 2.
	164C	8A C7	MOV AL,BH	Load MSB
	164E	EE	OUT DX,AL	
	164F	C3	RET	

FUNCTION NAME — ADC

INPUT — BL

OUTPUT — AL

CALLS — NONE

DESTROYS — AL,DX

DESCRIPTION — This routine initiates the ADC, addressing the channel no. by the content of reg. (BL), and inputs the digital equivalent of the analog voltage applied at the input of ADC into reg.(AL). The 24-channels are addressed by 00-17, as desoribed earlier.

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
	F000:1660	BA E6 FF	MOV DX,FFE6	PA and PC upper as o/p ports. PB and PC lower of 8255-I as I/P ports.
	1663	B0 83	MOV AL,83	
	1665	EE	OUT DX,AL	
	1666	BA E0 FF	MOV DX,FFE0	Send valid address and latch it through 8255
	1669	8A C3	MOV AL,BL	
	166B	EE	OUT DX,AL	
	166C	0C C0	OR AL,C0	Send ALE and start of conv. pulse (make high and then low). The address should remain valid throughout.
	166E	EE	OUT DX,AL	
	166F	8A C3	MOV AL,BL	
	1671	EE	OUT DX,AL	

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
	1672	BA E4 FF	MOV DX,FFE4	
L1	1675	EC	IN AL,DX	Check if EOC
	1676	A8 01	TEST AL,01	has gone low
	1678	75 FB	JNZ L1	
L2	167A	EC	IN AL,DX	Check if EOC
	167B	A8 01	TEST AL,01	has gone
	167D	74 FB	JZ L2	high
	167F	BA E2 FF	MOV DX,FFE2	Input the
	1682	EC	IN AL,DX	o/p of ADC
	1683	C3	RET	

FUNCTION NAME : FREQ

INPUT : Mem. locations 0000: 03F0-03F3

OUTPUT : CX

CALLS : NONE

DESTROYS : Al,BL,CX,DX

DESCRIPTION : This routine counts the no. of pulses in one sec at the PC_0 of 8255-II. Counter 0 of 8253 is loaded with a count of 61300_{10} in mode 0 for getting a delay of around 50 ms, which interrupts the processor through IR4 of 8259. When this interrupt comes twenty times, i.e. 50 x 20 ms = 1 sec are over the no. of pulses counted are returned in (CX). A description of this routine is given earlier. The offset and segment address for I.S.S. is contained in locations 0000:03F0-03F3

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
FREQ	F000:1690	BA FF FF	MOV DX,Cont.reg.	Initialize PC lower
	1693	BO 81	MOV AL,mode wd	as I/P port and
	1695	EE	OUT DX,AL	rest all as o/p port.
	1696	BA C8 FF	MOV DX, FFC8	
	1699	BO 13	MOV AL,XXX1 0X11	ICW1
	169B	EE	OUT DX, AL	Edge triggering
	169C	42	INC DX	
	169D	42	INC DX	
	169E	BO F8	MOV AL,111 1XXX	Type 252 interrupt
	16A0	EE	OUT DX, AL	at IR4

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
	16A1	B0 03	MOV AL,0000 0X11	8086 mode
	16A3	EE	OUT DX,AL	Automatic EOI not special fully nested mode.
	16A4	B0 CF	MOV AL,CF	Mask all interrupt
	16A6	EE	OUT DX,AL	lines except IR4
	16A7	B9 00 00	MOV CX,0000	Initial freq.count
	16AA	BA DE FF	MOV DX,Cont.reg.	Initialize counter 0
	16AD	B0 30	MOV Al,Mode wd	in mode 0, LSB first,
	16AF	EE	OUT DX,AL	MSB next,Binary counting.
	16B0	BA D8 FF	MOV DX,Counter 0	
	16B3	B0 74	MOV AL,74	Load a count
	16B5	EE	OUT DX,AL	$61300_{10}(EF74_{16})$
	16B6	B0 EF	MOV AL,EF	in counter 0
	16B8	EE	OUT DX,AL	
	16B9	B3 14	MOV BL,14	Count for no. of
	16BB	FB	STI	timer interrupts.
	16BC	BA FD FF	MOV DX,Port C	Poll through PCo of 8255-II
L1	16BF	EC	IN AL,DX	
	16C0	A8 01	TEST AL,01	Is PCo low?
	16C2	75 FB	JNZ L1	No,wait
L2	16C4	EC	IN AL,DX	Yes, poll again

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
	16C5	A8 01	TEST AL,01	Is PC_0 high ?
	16C7	74 FB	JZ L2	No, wait
	16C9	41	INC CX	Increment no.of pulses counted.
	16CA	EB F3	JMP L1	Branch for further counting
	16CC	C3	RET	Return

I.S.S. FOR IR 4

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
	F000:16D0	50	PUSH AX	Save registers
	16D1	52	PUSH DX	to be used.
	16D2	FE CB	DEC BL	20 interrupts over ?
	16D4	74 14	JZ L3	Yes branch
	16D6	BA DE FF	MOV DX,Cont.reg.	
	16D9	B0 30	MOV AL,mode wd	For
	16DB	EE	OUT DX,AL	further
	16DC	BA D8 FF	MOV DX,Counter 0	Interrupts
	16DF	B0 74	MOV AL,74	
	16E1	EE	OUT DX,AL	
	16E2	B0 EF	MOV AL,EF	
	16E4	EE	OUT DX,AL	
	16E5	5A	POP DX	Restore
	16E6	58	POP AX	registers
	16E7	FB	STI	
	16E8	EB 08	JMP L4	
L3	16EA	81 C4 06 00	ADD SP,0006	To point to CS of sub FREQ in the stack
	16EE	BB CC 16	MOV BX,Return add of sub.FREQ	
	16F1	53	PUSH BX	To point to return add of sub.FREQ.
L4	16F2	CF	IRET	Return from I.S.S.

ADDRESS TO SERVE IR4 I.S.S.
0000. 03F0 DB D0 16 00 F0

FUNCTION NAME — STAT

INPUT — Mem. locations
F000: 31B0 ~ 31B9

OUTPUT — Mem. location 000:3EE0 ~ 3EE1

CALLS — SETP, UPBBC

DESTROYS — AX,BX,CX,DX,SI,DI
Mem. locations 0000: 3EE0-3EE1,
0000: 3EF0-3EF1

DESCRIPTION — This routine asks for the ref. temp in BCD and converts this value into binary equivalent through subroutine UPBBC and stores in location 0000: 3EE0-3EE1.Then it jumps to subroutine STAT1 which monitors the temperature through PID control scheme.

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
STAT	F000:14B0	BB C0 11	MOV BX,SETP	
	14B3	FF D3	CALL SETP	Ask set point
	14B5	BB 00 00	MOV BX,0000	(DS) ← 0000
	14B8	8E DB	MOV DS,BX	
	14BA	BB 00 18	MOV BX,UPBBC	
	14BD	FF D3	CALL UPBBC	Convert into binary equiv.
	14BF	BB 00 17	MOV BX,STAT1	Monitor variable
	1462	FF E3	JMP STAT 1	

FUNCTION NAME: STAT 1

INPUT: Mem. locations 0000:03F4-03F7, 3F00-3F27

OUTPUT: None

CALLS: ADO, PID, SCR, SAPPD

DESTROYS: AX, BX, CX, DX, SI, DI, BP, DS
Mem. locations 0000: 3F00-3F1D, 3F28-3F2F

DESCRIPTION: This routine monitors the temperature through PID control scheme. The fifteen errors (assumed zero initially) are stored in locations 0000:3F00-3F1D; parameters K in 0000: 3F1E-3F1F, $K_I T_S$ in 0000:3F20-3F21, $K_D T_S$ in 0000:3F22-23. The value of P_S is to be stored in 0000:3F24-3F27 before executing the main program.

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
STAT1	F000:1700	BB 00 00	MOV BX,0000	
	1703	8E DB	MOV DS,BX	(DS) ←0000
	1705	BC 00 01	MOV SP,0100	(SP) ←0100
	1708	BA DE FF	MOV DX,FF DE	Initialize 8253, counter
	170B	BO 76	MOV AL,76	1 in mode 3
	170D	EE	OUT DX,AL	
	170E	BA DA FF	MOV DX,FF DA	
	1711	BO 74	MOV AL,74	Load count for time
	1713	EE	OUT DX, AL	period of 50 ms
	1714	BO EF	MOV AL,EF	in counter 1
	1716	EE	OUT DX, AL	

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
	1717	BD A0 00	MOV BP, 00 A0	No. of interrupts of duration 50 ms each, for a sampling pol of 8 sec.
	171A	BA C8 FF	MOV DX,FFC8	Initialize 8259
	171D	B0 13	MOV AL,13	Edge triggering
	171F	EE	OUT DX,AL	
	1720	42	INC DX	
	1721	42	INC DX	
	1722	B0 F8	MOV AL,F8	Type 253 interrupt
	1724	EE	OUT DX,AL	at IR5
	1725	B0 03	MOV AL,03	8086 mode, AEOL,
	1727	EE	OUT DX,AL	Not special fully nodec mode
	1728	B0 DF	MOV AL,DF	Unmask IR5
	172A	EE	OUT DX,AL	
	172B	FC	CLD	Auto increment
	172C	90	NOP	
	172D	90	NOP	
	172E	90	NOP	
	172F	FB	STI	Enable interrupt
L1	1730	F4	HALT	Synchronise with 8 sec.sampling interval
	1731	EB FD	JMP L1	
	1733	BD A0 00	MOV BP,00A0	Reload counter for 160 times 50 ms interrupts

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
	1736	B3 11	MOV B1,11	Input digital equiv.
	1738	BE 60 16	MOV SI,1660	of temp through
	173B	FF D6	CALL ADC	Channel 1 of ADC-3 in reg. AL
	173D	B4 00	MOV AH,00	Higher byte ← 00
	173F	BE E0 3E	MOV SI,3EE0	
	1742	8B 14	MOV DX,(SI)	(DX)←Set point
	1744	2B D0	SUB DX,AX	(DX) ←Error
	1746	BF C 3F	MOV DI,3F C	Store error in
	1749	89 15	MOV (DI),DX	0000:3FIC-3FID
	174B	BE 80 17	MOV SI,PID	
	174E	FF D6	CALL PID	(DX)(AX)←$160^{\circ}-\alpha$
	1750	F7 C2 00 80	TEST DX,8000	Is $(160^{\circ}-\alpha)<0$?
	1754	74 05	JZ L2	No, branch
	1756	BB 89 2A	MOV BX,2A89	Yes, set firing angle← 2A89 (i.e. 160°)
	1759	EB 1C	JMP L3	Branch
L2	175B	F7 C2 FF 7F	TEST DX,7FFF	Is $(160^{\circ}-\alpha)>255$?
	175F	75 13	JNZ L4	Yes, branch
	1761	BB A0 00	MOV BX,160_{10}	No., evaluate α
	1764	2B D8	SUB BX,AX	(BX)← α in degrees
	1766	78 0C	JS L4	Branch if $\alpha < 0^{\circ}$
	1768	B0 44	MOV AL,68_{10}	Timer count ←
	176A	F6 E3	MUL AL,BL	α(degrees)X68_{10}

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
	176C	0B C0	OR AX,AX	Is count zero ?
	176E	74 04	JZ L4	Yes,branch
	1770	8B D8	MOV BX,AX	(BX) ← count
	1772	EB 03	JMP L3	Branch
L4	1774	BB 01 00	MOV **BX 0001**	0^{o}firing angle count
L3	1777	BE 40 16	MOV SI,SCR	Issue firing pulses
	177A	FF D6	CALL SCR	to SCR bridge
	177C	EB B2	JMP L1	Wait till next sampling period.

ADDRESS TO SERVE IRS,i.e.SAPPD :

0000:03F4 DB F0 17 00 F0

FUNCTION NAME PID

INPUT Mem locations 0000:3F00- 3F27

OUTPUT AX, DX

CALLS None

DESCTROYS AX,BX,CX,DX,SI,DI

Mem. locations 0000: 3F00-3F1D, 3F28-2F2F

DESCRIPTION This routine, through PID control scheme calculates the actuating signal value in 4 bytes and returns the result in (DX)(AX), with higher bytes in DX and lower bytes in AX.

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
PID	F000:1780	B1 0E	MOV CL,0E	No. of additions of error
	1782	BE 00 3F	MOV SI,3F00	
	1785	AD	LODS W	
L1	1786	03 04	ADD AX,(SI)	
	1788	46	INC SI	
	1789	46	INC SI	
	178A	FE C9	DEC CL	$(AX) \leftarrow \sum_{i=n-14}^{n} e_i$
	178C	75 F8	JNZ L1	
	178E	BE 20 3F	MOV SI,3F20	
	1791	F7 2C	IMUL AX,(SI)	Integral term in (DX)(AX)
	1793	BF 28 3F	MOV DI,3F 28	Store integral term
	1796	AB	STOS W	in 3F28-3F2B
	1797	89 15	MOV(DI),DX	

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
	1799	BE 1A 3F	MOV SI, 3F 1A	
	179C	AD	LODS W	
	179D	8B 14	MOV DX,(SI)	
	179F	2B D0	SUB DX,AX	(DX)← $e_n - e_{n-1}$
	17A1	89 D0	MOV AX,DX	(AX)← $e_n - e_{n-1}$
	17A3	BE 22 3F	MOV SI,3F22	
	17A6	F7 2C	IMUL AX,(SI)	Derivative term in
	17A8	47	INC DI	(DX)(AX)
	17A9	47	INC DI	
	17AA	AB	STDS W	Store derivative term
	17AB	89 15	MOV (DI),DX	in 3F2C-3F2F
	17AD	BE 1C 3F	MOV SI,3F1C	
	17B0	AD	LODS W	
	17B1	F7 2C	IMUL AX,(SI)	(AX)← K. e_n
	17B3	8B C8	MOV CX,AX	(CX)← K. e_n
	17B5	B3 03	MOV BL,03	No.of terms to be added
	17B7	BE 24 3F	MOV SI,3F34	
L2	17BA	AD	LODS W	
	17BB	03 C8	ADD CX,AX	
	17BD	AD	LODS W	

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
	17BE	13 D0	ADC DX,AX	(DX)(CX)←P+I+D terms
	17C0	FE CB	DEC BL	+Ps
	17C2	75 F6	JNZ L2	
	17C4	89 C8	MOV AX,CX	Result in (DX)(AX)
	17C6	B9 0E 00	MOV CX,000E	No. of transfers of errors
	17C9	BF 00 3F	MOV DI,3F00	$e_{i-1} \leftarrow e_i (i=n-13, n)$
	17CC	BE 02 3F	MOV SI,3F02	
	17CF	F2 A5	REP MOVSW	
	17D1	C3	RET	

FUNCTION NAME : SAPPD

INPUT : NONE

OUTPUT : NONE

CALLS : NONE

DESTROYS : NONE

DESCRIPTION : This routine is used as an Interrupt Service Routine for IR5 to indicate the beginning of a sampling pds. of 8 sec. The count in reg. BP is initialized to 160. This routine is serviced after every 50 ms. When it is executed 160 times, the routine returns to monitor temperature in subroutine STAT1.

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
SAPPD	17E0	53	PUSH BX	Save reg.
	17E1	4D	DEC BP	Decrement Counter
	17E2	75 0A	JNZ L1	Is counter zero ?
	17E4	81 C4 04 00	ADD SP,0004	Yes,
	17E8	BB 33 17	MOV BX,1733	Monitor temp in
	17EB	53	PUSH BX	next sampling pd.
	17EC	4C	DEC SP	
	17ED	4C	DEC SP	
L1	17EE	5B	POP BX	Restore reg.
	17EF	FB	STI	Eneble interrupt
	17F0	CF	IRET	

FUNCTION NAME : UPBBC

INPUT : Location 0000: 3EF0-3EF1

OUTPUT : Location 0000: 3EE0-3EE1

CALLS : None

DESTROYS : AX,CL,SI,DS
Location 0000: 3EE0-3EE1

DESCRIPTION : This routine converts the ref. temp. value stored in unpacked BCD format in 0000: 3EF0-3EF1 into a digital equivalent which would have been obtained as an o/p of the ADC.

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
UPBBC	F000: 1800	FC	CLD	Auto increment
	1801	BE F0 3E	MOV SI, 3EF0	(SI)←address of unpacked BCD no.s
	1804	AD	LODS W	(AX)←Unpacked BCD value of temp.
	1805	B1 04	MOV CL,04	Shift left higher byte
	1807	D2 E4	SHL AH,04	four times
	1809	0A C4	OR AL,AH	(AL)←packed BCD
	180B	B4 00	MOV AH,00	
L1	180D	FE C4	INC AH	Convert to
	180F	2C 01	SUB AL,01	binary in
	1811	2F	DAS	AH
	1812	75 F9	JNZ L1	
	1814	8A C4	MOV AL,AH	
	1816	B1 CC	MOV CL,CC	
	1818	F6 E1	MUL AL,CL	S←Ref.value * 204_{10}

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
	181A	B1 64	MOV CL,64	
	181C	F6 F1	DIV AX,CL	Digital equiv. $\leftarrow S/100_{10}$
	181E	B4 00	MOV AH,00	Higher byte $\leftarrow$(00)
	1820	BF E0 3E	MOV DI,3E E0	Store digital equiv.
	1823	AB	STDSW	in 0000:3EE0-1
	1824	C3	RET	Return

FUNCTION NAME : DTRF

INPUT : LOCATIONS F000: 3200-320D

OUTPUT : LOCATIONS 0000: 03F0-03F7,3F00-3F23

CALLS : NONE

DESTROYS : AX,BX,CX,SI,DI
LOCATIONS 0000:03F0-03F7,3F00-3F23

DESCRIPTION : This routine transfers the data to be used by many routines from EPROM to RAM area. Thus, every time the power is put on, this data is not to be stored in RAM area before executing the main program.

LEBEL	ADDRESS	CONTENTS	MNEMONICS AND OPERANDS	COMMENTS
DTRF	F000:1830	BB 00 F0	MOV BX,F000	
	1833	8E DB	MOV DS,BX	(DS)← F000
	1835	BE 00 32	MOV SI,3200	Source ←F000:3200
	1838	BF F0 03	MOV DI,03F0	Destination 0000:03F0
	183B	B9 04 00	MOV CX,0004	Transfer offset and
	183E	FC	CLD	Segment addresses
	183F	F2 A5	REP MOVSW	for IR4 and IR5 I.S.S.
	1841	B8 00 00	MOV AX,0000	Move zero to all
	1844	BF 00 3F	MOV DI,3F00	the errors e_{n-14} to
	1847	B9 0F 00	MOV CX,000F	e_n
	184A	F2 AB	REP STOSW	Transfer PID control
	184C	B9 03 00	MOV CX,0003	parameters to
	184F	F2 A5	REP MOV SW	0000: 3F1E-3F23
	1851	C3	RET	Return

CONTENTS TO BE TRANSFERRED:

ASSUME DS: F000

F000:3200 DB D0 16 00 F0 E0 17 00 F0 02 00 05 00 01 00

CHAPTER - VII

EXPERIMENTATION AND RESULTS

7.1 EXPERIMENT PROCEDURE

The experimental set up is assembled as shown in Fig. 2.3.1 and the process to be controlled in Fig.2.1.1 (chapter II).

Before conducting experiment, the whole set up is cleaned for the scale deposits by using solution of caustic soda followed by water wash and then dilute acid wash and finally with distilled water till neutral wash is obtained.

7.1.1 OPERATING PROCEDURE

First of all the water inlet (14) to the overhead tank (1) is opened. The flow rate in the tank is maintained in such a way that the overflow line (2) starts functioning. The valve (3) is opened to allow water into the CSTR (7). The control valve (12) is kept at full open condition to drain the water. The valve (3) is opened to such an extent that a constant level develops in the CSTR. The level in all the cases should atleast be 6 cms.more than the height of the immersion heaters. The stirrer (6) and the main heater (8) are activated simultaneously. After around 17 minutes, the temperature in the CSTR becomes steady.

7.1.2 GENERATION OF STEP INPUT

The Step input to the CSTR can be given in two ways- either by introducing a step change in load variable (liquid flow rate to CSTR or its temperature or heat developed by main heaters) or in set point. The magnitude of step input (in load variable) can be controlled by regulating the valve (4) and then energising the solenoid valve (5). This scheme controls the auxiliary liquid fluid to CSTR.

The step change in set point can be generated with the help of μp.

7.1.3 EXPERIMENTATION

1) VOLTAGE AND POWER VS FIRING ANGLE FOR SCR CONVERTER BRIDGE:

The voltage and power delivered to the auxiliary heater vs. firing angle characterstics generated for SCR bridge were recorded, as tabulated in Appendix A and plotted, as shown in Fig. 7.1.1. As can be noted from this characterstic, the power delivered at 160° firing angle almost becomes zero. This is used for all calculation purposes.

2) PID CONTROL OF CSTR

The CSTR was operated as described in sec.7.1.1 and a constant level of water was maintained. The flow rate of water into the system was kept at 1.8 LPM. The main heaters were put on at 220V and the temperature time history of the

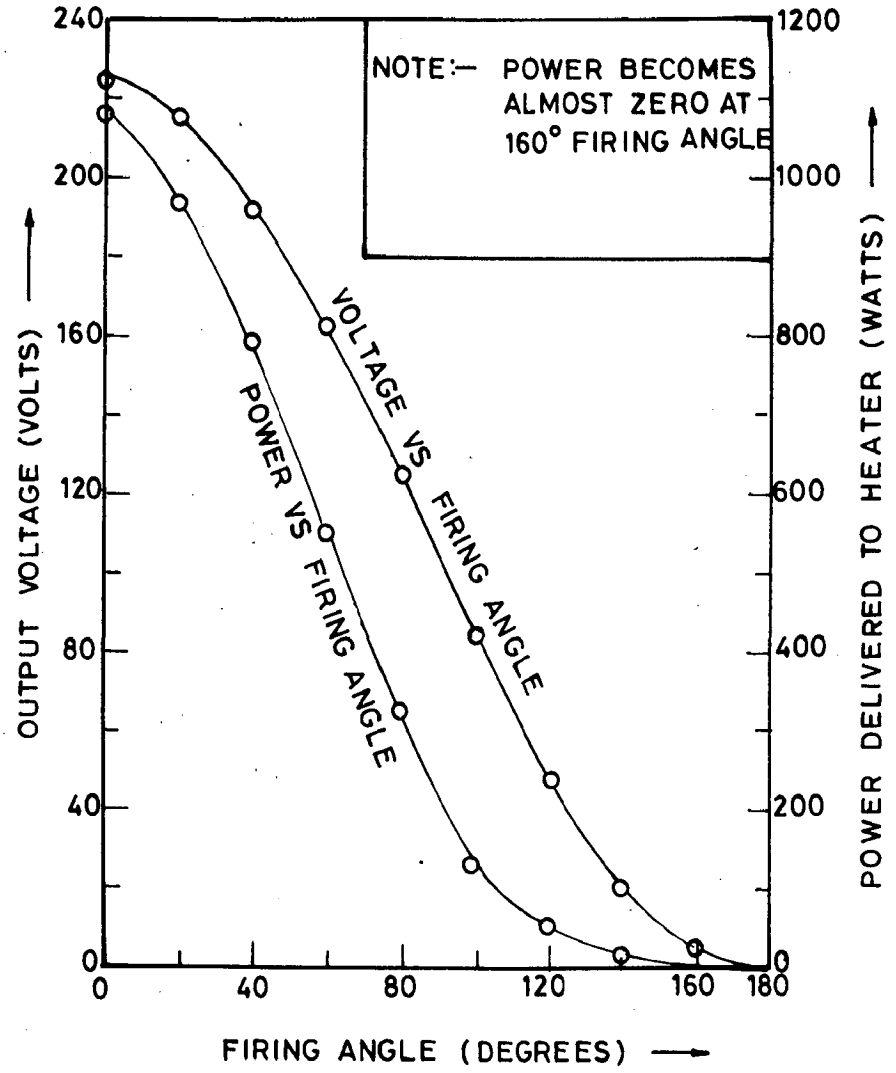


FIG. 7.1-1 POWER AND VOLTAGE VS FIRING ANGLE CHARACTERISTICS OF CONVERTER BRIDGE

CSTR was noted upto the steady state. After the steady state was achieved, the set point was fixed at 40°C and the PID control algorithm was executed for a set of control parameters K, K/T_I and KT_D values. The time-temperature history of the process was noted upto a fresh steady state under the PID control scheme. Then the control heater was put off and the system was allowed to reach its original steady state value. This was necessary to start a fresh run with different set of control parameters. This sequence was repeated for PID control with different values of control parameters, out of which results of only three sets are presented.

The different sets of $K, K/T_I$ and KT_D values along with different operating conditions under which the experiments were conducted are given in Table 7.1.

TABLE 7.1 OPERATING PARAMETERS

Set No.	K	K_I T_s	K_D/T_s	T_s (sec.)
1	2	5	1	8
2	10	5	2	8
3	10	5	2	4

Liquid Flow Rate : 1.8 LPM
Supply to Main Heater: 220 volts

By changing the control parameters by trial and error, an optimum combination was obtained as reported in set no.3.

7.2 RESULTS AND DISCUSSIONS

The results of the experimentation as described in sec. 7.1.3 were recorded and normalised are reported in Appendix A. These results are plotted in Fig. 7.2.1. From the plot in Fig. 7.2.1, the time constant τ of the CSTR comes out to be 4.5 minutes.

Out of around twenty experiments conducted, covering different values of the control parameters for a sampling period of 8 sec., it was observed that $K = 10$, $K_I T_S = 5$, $K_D/T_S = 2$ gave the best results.

In the next set of experiments it was planned to check the effect of the sampling time on the mode of control. It was thought necessary to sample temperature shoot-ups due to system fluctuations by decreasing the sampling time. The sampling time was cut-off to 4 secs. It is clear from the Fig.7.2.1, plot 4 that the concept gave a better result. In terms of actual temperature, a fluctuation of $\pm$ 0.25^{o}C was observed around the set point by doing so. However, further decrease in sampling time moved the system towards on/off control.

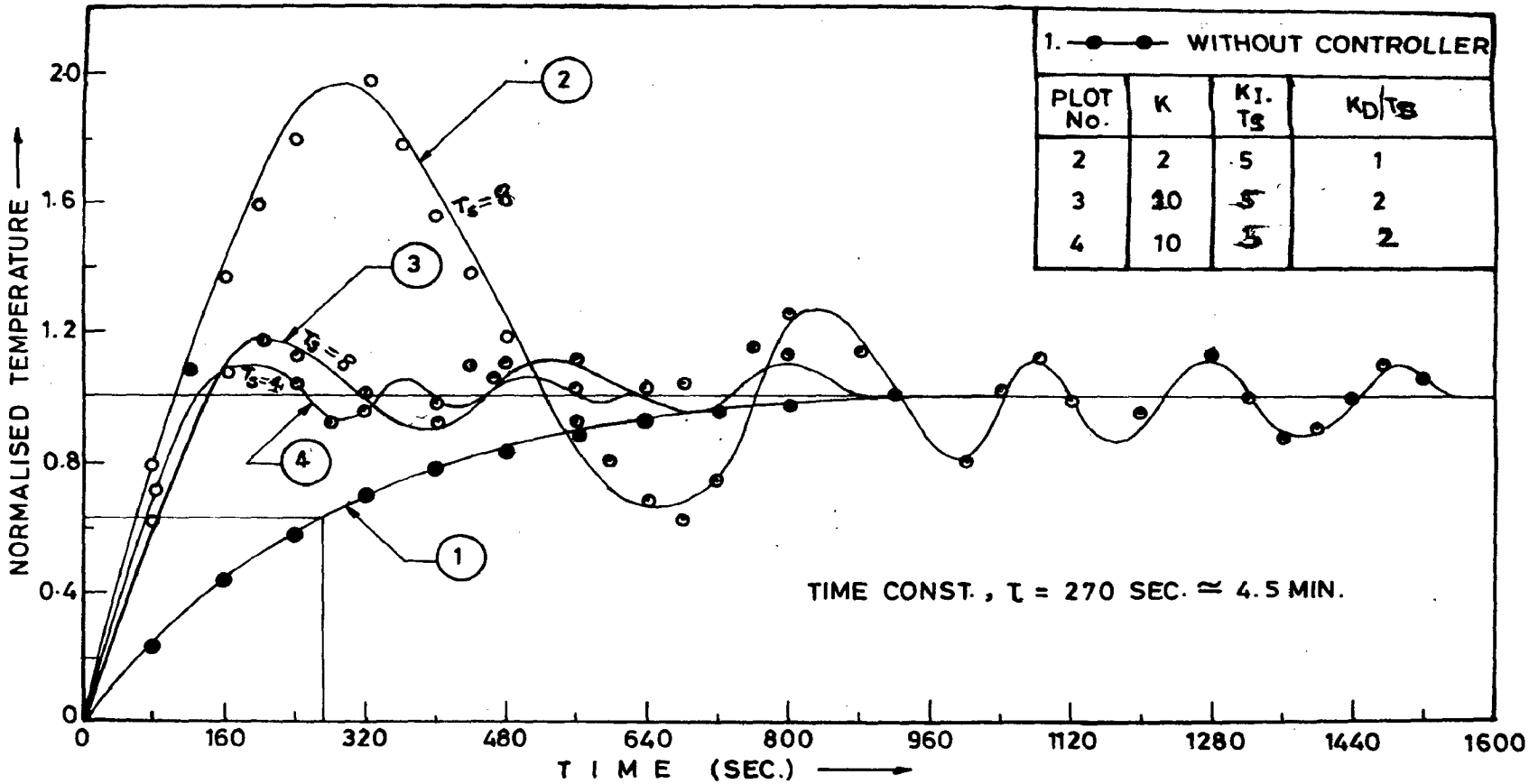


FIG. 7.2-1 TEMPERATURE RESPONSE WITH TIME

CHAPTER - VIII

CONCLUSIONS AND SUGGESTIONS FOR FURTHER DEVELOPMENTS

The conclusions and suggestions emerged out of the present work are listed below:

CONCLUSIONS

From the experiments conducted and the results analysed, the following conclusions can be made:

1. The 8086 can proved to be a very powerful μp for the control applications.
2. The temperature of the CSTR can be very effectively controlled by PID control scheme through the μp.
3. A very cost effective and accurate capacitive level transducer can be developed with the help of highly linear C/F converter as developed in the present work.

RECOMMENDATIONS

1. Multivariable combined control scheme for level and flow along with temperature should be taken to utilize the potentiality of 8086 μp.
2. To increase the accuracy for temperature control, a 12 bit A/D converter should be used.
3. A control scheme for self-tuning the control parameters should be used.

4. An arrangement should be made to get feed - back from the motor at outlet of the CSTR about its angle of rotation for controlling the level.

REFERENCES

[1] User's Manual, VMC-86/3 Microprocessor Training/ Development Kit, Vinytics Peripherals Pvt.Ltd., Delhi.

[2] Intel's Microsystem Components Handbook, Vol. I and II, 1984.

[3] Intel's iAPX 86,88 User's Manual, 1981.

[4] Gibson and Liu, Microcomputer Systems: The 8086/8088 Family, Prentice-Hall of India Publication.

[5] I.C.Master, Vol. I and II, 1983.

[6] Texas Instruments 'The TTL Data Book for Design Engineers', 1981.

[7] Bibbero, Robert J., 'Microprocessors in Instruments and Control', Wiley-Interscience Publication, 1977.

[8] Russell Rector and George Alexy 'The 8086 Book', Osborne/McGraw Hill Publication.

[9] Cohen, G.H. and Coon, G.A., Trans. ASME, 75: 827 (1953).

[10] Coughanowr, Donald R. and Koppel, Lowell B., 'Process Systems Analysis and Control'.

[11] Ross, P.J., 'A Water-Level Sensor using a capacitance to frequency converter', Journal of Physics, E:Scientific Instrument, Vol. 16, Aug. 1983, pp. 827.

[12] Hall, Dougles V., 'Microprocessors in Instruments and Control', McGraw-Hill Publication.

[13] Dewan, S.B. and Stranghen, A., 'Power Semiconductor Circuits', Wiley- Interscience Publication, 1975.

[14] Krikelis, Nicholas J. and Fassois, Spilios D., 'Microprocessor Implementation of PID controllers and Lead Lag Compensators', I.E.E.E.Transactions on Industrial Electronics, Vol. IE-31, No.1, Feb., 1984.

[15] 'Microcomputer Data Handbook', Business Promotion Bureau, Delhi, First Edition, 1983.

[16] 'Practical Semiconductors Data Mannual', Volume-2, Business Promotion Bureau, Delhi, 1976.

[17] 'Stepper Motor Interface', Vinytics, Delhi.

APPENDIX A

EXPERIMENTAL RESULTS

1) VARIATION OF CONVERTER BRIDGE O/P WITH FIRING ANGLE

S.No.	FIRING ANGLE (DEGREES)	TIMER COUNT	OUTPUT VOLTAGE (VOLTS)	OUTPUT CURRENT (AMP)	POWER DELIVERED (WATTS)
1.	0	$0001_{10}=0001_{16}$	225	4.8	1080
2.	10	$680_{10}=02A8_{16}$	222	4.75	1054.5
3.	20	$1361_{10}=0551_{16}$	215.5	4.5	969.75
4.	30	$2042_{10}=07FA_{16}$	206.2	4.3	886.66
5.	40	$2722_{10}=0AA2_{16}$	193.2	4.1	792. 12
6.	50	$3403_{10}=0D4B_{16}$	180	3.8	684
7.	60	$4083_{10}=0FF3_{16}$	163	3.4	554.2
8.	70	$4764_{10}=129C_{16}$	144	3	432
9.	80	$5444_{10}=1544_{16}$	125.5	2.6	326.3
10.	90	$6125_{10}=17ED_{16}$	104.5	2.1	219.45
11.	100	$6805_{10}=1A95_{16}$	84.0	1.7	128
12.	110	$7486_{10}=1D3E_{16}$	65.5	1.25	87.875
13.	120	$8167_{10}=1FE7_{16}$	48	1	48
14.	130	$8847_{10}=228F_{16}$	32.4	0.7	22.68
15.	140	$9528_{10}=2538_{16}$	20	0.4	8
16.	150	$10208_{10}=27E0_{16}$	9.5	0.2	1.9
17.	160	$10889_{10}=2A89_{16}$	3.75	0.08	0.3
18.	170	$11569_{10}=2D31_{16}$	0	0	0
19.	180	$11250_{10}=2FDA_{16}$	0	0	0

Resistance of Heater $\simeq$ 47 ohms.

These results are plotted in Fig. 7.1.1.

2) STEP RESPONSE OF TEMPERATURE WITH TIME WITHOUT CONTROLLER

Step I/P - 220V, 10.5 A

S.No.	TIME (SEC)	THERMOCOUPLE E.M.F.(mv)	TEMP. (oC) Ti	NORMALISED TEMP.
1.	0	0.922	23.45	0
2.	20	0.949	24.117	
3.	40	0.989	25.102	0.108
4.	60	1.028	26.062	
5.	80	1.059	26.825	0.220
6.	100	1.088	27.539	
7.	120	1.118	28.277	0.315
8.	140	1.152	29.114	
9.	160	1.177	29.729	0.410
10.	180	1.029	30.51	
11.	200	1.221	30.801	0.480
12.	220	1.245	31.384	
13.	240	1.267	3o.918	0.554
14.	260	1.288	32.427	
15.	280	1.305	32.840	0.614
16.	300	1.326	33.350	
17.	320	1.346	33.835	0.679
18.	340	1.363	34.248	
19.	360	1.385	34.782	0.741
20.	380	1.391	35.073	
21.	400	1.424	35.728	0.803

S.No.	TIME (SEC)	THERMOCOUPLE E.M.F.(mv)	TEMP. (oC)Ti	NORMALISED TEMP.
22.	420	1.424	35.728	
23.	440	1.438	36.068	0.825
24.	460	1.446	36.262	
25.	480	1.457	36.529	0.855
26.	500	1.473	36.918	
27.	520	1.479	37.063	0.890
28.	540	1.482	37.136	
29.	560	1.496	37.476	0.917
30.	580	1.503	37.646	
31.	600	1.508	37.767	0.936
32.	620	1.514	37.913	
33.	640	1.522	38.107	0.959
34.	660	1.528	38.252	
35.	680	1.536	38.447	0.981
36.	700	1.548	38.738	
37.	720	1.539	38.519	0.985
38.	740	1.545	38.665	
39.	760	1.549	38.714	0.998
40.	780	1.547	38.738	
41.	800	1.547	38.738	1

Initial Temp = 23.45°C

Steady State Temp. = 38.738 °C

Difference between steady state value and initial value,

$$\Delta T = 38.738 - 23.45 = 15.288$$

Sample calculation for Normalised temperature:

For S.No.3,

$$\text{Normalised Temp} = \frac{T_i - \text{Initial Temp}}{\Delta T}$$

$$= \frac{25.102 - 23.45}{15.288}$$

$$= 0.108$$

These results are plotted in Fig. 7.2.1, plot 1.

3) TEMPERATURE RESPONSE WITH CONTROLLER

Main Heater Supply : 220V

Water Flow Rate : 1.8 Litres/Min.

Reference Temp : 40°C

i) $K = 2$, $K_I T_S = 5$, $K_D/T_S = 1$, $T_S = 8$ sec.

S.No.	TIME (SEC)	THERMOCOUPLE E.M.F.(mv)	TEMPERATURE (°C) T_i	NORMALISED TEMP
1.	0	1.505	37.694	0
2.	40	1.544	38.641	
3.	80	1.579	39.49	0.778
4.	120	1.609	40.214	
5.	160	1.636	40.857	1.372
6.	200	1.660	41.429	
7.	240	1.68	41.905	1.826
8.	280	1.698	42.333	
9.	320	1.694	42.238	1.97
10.	360	1.676	41.81	
11.	400	1.654	41.286	1.557
12.	440	1.635	40.833	
13.	480	1.619	40.452	1.196
14.	520	1.604	40.095	
15.	560	1.592	39.806	0.915
16.	600	1.581	39.539	
17.	640	1.571	39.296	0.694

S.No.	TIME (SEC)	THERMOCOUPLE E.M.F.(mv)	TEMPERATURE (oC) T_i	NORMALISED TEMP
18.	680	1.563	39.102	
19.	720	1.556	38.932	0.537
20.	760	1.576	39.903	
21.	800	1.63	40.714	1.26
22.	840	1.628	40.667	
23.	880	1.613	40.143	1.13
24.	920	1.599	39.976	
25.	960	1.587	39.685	0.87
26.	1000	1.577	39.442	
27.	1040	1.606	40.143	1.02
28.	1080	1.612	40.286	
29.	1120	1.598	39.952	0.98
30.	1160	1.587	39.675	
31.	1200	1.597	39.927	0.96
32.	1240	1.606	40.143	
33.	1280	1.614	40.333	1.13
34.	1320	1.6	40.000	
35.	1360	1.588	39.709	0.87
36.	1400	1.591	39.782	
37.	1440	1.601	40.024	1
38.	1480	1.61	40.238	
39.	1520	1.607	40.167	1.06

Initial Temperature = 37.694

Steady State Temp = 40°C

ΔT = 40-37.694 = 2.306

Sample calculation for Normalised Temperature:

For S.No.3,

$$\text{Normalised Temp} = \frac{T_i - \text{Initial Temp}}{\Delta T}$$

$$= \frac{39.49 - 37.694}{2.306}$$

$$= 0.778$$

These results are plotted in Fig. 7.2.1, Plot No.2,

ii) $K = 20$, $K_I T_S = 5$, $K_D / T_S = 2$

S.No	TIME (SEC)	THERMOCOUPLE E.M.F.(mv)	TEMPERATURE (°C), Ti	NORMALISED TEMP
1.	0	1.414	35.485	0
2.	40	1.479	37.063	0.35
3.	80	1.527	38.228	0.607
4.	120	1.580	39.515	0.89
5.	160	1.622	40.524	1.12
6.	200	1.630	40.714	1.16
7.	240	1.620	40.476	1.107
8.	280	1.619	40.452	1.102
9.	320	1.598	39.952	0.99
10.	360	1.584	39.612	0.914
11.	400	1.583	39.587	0.908
12.	440	1.618	40.429	1.096
13.	480	1.636	40.857	1.19
14.	520	1.636	40.857	1.19
15.	560	1.632	40.762	1.17
16.	600	1.617	40.405	1.09
17.	640	1.606	40.143	1.03
18.	680	1.607	40.167	1.037
19.	720	1.595	39.879	0.97
20.	760	1.629	40.691	1.156
21.	800	1.627	40.643	1.145
22.	840	1.608	40.191	1.043

Initial Temperature $= 35.485^{\circ}C$

Steady State Temp. $= 40^{\circ}C$

$\Delta T = 40 - 35.485 = 4.515^{\circ}C$

iii) K = 10, $K_I T_S = 5$, $K_D/T_S = 2$, $T_s = 4$ sec.

S.No.	TIME SEC	THERMOCOUPLE E.M.F.(mv)	TEMPERATURE (°C), Ti	NORMALISED TEMP
1.	0	1.467	36.772	0
2.	20	1.495	37.452	
3.	40	1.516	37.961	0.368
4.	60	1.544	38.641	
5.	80	1.572	39.320	0.789
6.	100	1.580	39.515	
7.	120	1.590	39.757	0.924
8.	140	1.598	39.952	
9.	160	1.611	40.262	1.081
10.	180	1.608	40.191	
11.	200	1.615	40.357	1.11
12.	220	1.620	40.476	
13.	240	1.606	40.143	1.044
14.	260	1.592	39.806	
15.	280	1.588	39.709	0.909
16.	300	1.590	39.757	
17.	320	1.595	39.879	0.962
18.	340	1.600	40.00	
19.	360	1.605	40.119	1.036
20.	380	1.603	40.071	
21.	400	1.596	39.903	0.969
22.	420	1.602	40.048	

S.No.	TIME SEC	THERMOCOUPLE E.M.F.(mv)	TEMPERATURE (oC),Ti	NORMALISED TEMP
23.	440	1.602	40.048	1.014
24.	460	1.609	40.214	
25.	480	1.614	40.333	1.103
26.	500	1.620	40.476	
27.	520	1.612	40.268	1.083
28.	540	1.611	40.262	
29.	560	1.605	40.119	1.036
30.	580	1.599	39.976	

Initial Temperature = 36.772^{o}C

Steady State Temp. = 40^{o}C

ΔT = 40 - 36.772 = 3.228^{o}C.

APPENDIX B

DETAILS OF PIN ASSIGNMENT ON MULTIBUS AND CONNECTORS

1. PIN ASSIGNMENT OF BUS SIGNALS ON MULTI-BUS BOARD CONNECTOR:

1.	-	GND	26.	-	NC
2.	-	GND	27.	-	$\overline{\text{BHEN}}$
3.	-	+5V	28.	-	MA16
4.	-	+5V	29.	-	$\overline{\text{CBRQ}}$
5.	-	+5V	30.	-	MA17
6.	-	+5V	31.	-	$\overline{\text{CCLK}}$
7.	-	+12V	32	-	$\overline{\text{MA18}}$
8.	-	+12V	33.	-	$\overline{\text{SINTA}}$
9.	-	-5V	34.	-	$\overline{\text{MA19}}$
10.	-	-5V	35.	-	$\overline{\text{INT6}}$
11.	-	GND	36.	-	$\overline{\text{INT7}}$
12.	-	GND	37.	-	$\overline{\text{INT4}}$
13.	-	$\overline{\text{BCLK}}$	38.	-	$\overline{\text{INT5}}$
14.	-	$\overline{\text{INIT}}$	39.	-	NC
15.	-	$\overline{\text{BPRN}}$	40.	-	$\overline{\text{INT3}}$
16.	-	$\overline{\text{BPRQ}}$	41.	-	NC
17.	-	$\overline{\text{BUSY}}$	42.	-	NC
18.	-	$\overline{\text{BREQ}}$	43.	-	$\overline{\text{MA14}}$
19.	-	$\overline{\text{SMRD}}$	44.	-	$\overline{\text{MA15}}$
20.	-	$\overline{\text{SMWR}}$	45.	-	$\overline{\text{MA12}}$
21.	-	$\overline{\text{SIRD}}$	46.	-	$\overline{\text{MA13}}$
22.	-	$\overline{\text{SIWR}}$	47.	-	$\overline{\text{MA10}}$
23.	-	$\overline{\text{XACK}}$	48.	-	$\overline{\text{MA11}}$
24.	-	NC	49.	-	$\overline{\text{MA8}}$
25.	-	NC	50.	-	$\overline{\text{MA9}}$

51.	-	$\overline{MA6}$	79.	-	-12V
52.	-	$\overline{MA7}$	80.	-	-12V
53.	-	$\overline{MA4}$	81.	-	+5V
54.	-	$\overline{MA5}$	82.	-	+5V
55.	-	$\overline{MA2}$	83.	-	+5V
56.	-	$\overline{MA3}$	84.	-	+5V
57.	-	$\overline{MA0}$	85.	-	GND
58.	-	$\overline{MA1}$	86.	-	GND
59.	-	$\overline{SD14}$			
60.	-	$\overline{SD15}$			
61.	-	$\overline{SD12}$			
62.	-	$\overline{SD13}$			
63.	-	$\overline{SD10}$			
64.	-	$\overline{SD11}$			
65.	-	$\overline{SD8}$			
66.	-	$\overline{SD9}$			
67.	-	$\overline{SD6}$			
68.	-	$\overline{SD7}$			
69.	-	$\overline{SD4}$			
70.	-	$\overline{SD5}$			
71.	-	$\overline{SD2}$			
72.	-	$\overline{SD3}$			
73.	-	$\overline{SD0}$			
74.	-	$\overline{SD1}$			
75.	-	$\overline{GND}$			
76.	-	$\overline{GND}$			
77.	-	NC			
78.	-	NC			

2) DETAILS OF CONNECTOR J1

The 24 I/O Lines of 8255-1 are brought out at this connector.

Pin	Signal	Pin	Signal
1.	PIC4	14.	P1B1
2.	P1C5	15.	P1A6
3.	P1C2	16.	P1A7
4.	P1C3	17.	P1A4
5.	P1C0	18.	P1A5
6.	P1C1	19.	P1A2
7.	P1B6	20.	P1A3
8.	P1B7	21.	P1A0
9.	P1B4	22.	P1A1
10.	P1B5	23.	P1C6
11.	P1B2	24.	P1C7
12.	P1B3	25.	GND
13.	P1B0	26.	GND

3) DETAILS OF CONNECTOR J2

1.	P2C4	14.	P2B1
2.	P2C5	15.	P2A6
3.	P2C2	16.	P2A7
4.	P2C3	17.	P2A4
5.	P2C0	18.	P2A5
6.	P2C1	19.	P2A2
7.	P2B6	20.	P2A3
8.	P2B7	21.	P2A0
9.	P2B4	22.	P2A1
10.	P2B5	23.	P2C6
11.	P2B2	24.	P2C7
12.	P2B3	25	GND
13.	P2B0	26.	GND

4) DETAILS OF CONNECTOR J3

Pin	Signal	Pin	Signal
1.	P3C4	14.	P3B1
2.	P3C5	15.	P3A6
3.	P3C2	16.	P3A7
4.	P3C3	17.	P3A4
5.	P3C0	18.	P3A5
6.	P3C1	19.	P3A2
7.	P3B6	20.	P3A3
8.	P3B7	21.	P3A0
9.	P3B4	22.	P3A1
10.	P3B5	23.	P3C6
11.	P3B2	24.	P3C7
12.	P3B3	25.	GND
13.	P3B0	26.	GND

5) DETAILS OF CONNECTOR J4

Pin	Signal	Pin	Signal
1.	GND	8.	NC
2.	Vcc	9.	-12V
3.	GND	10.	+12V
4.	Vcc	11.	NC
5.	Vcc(CM OS)	12.	24V
6.	GND	13.	NC
7.	NC		

6) DETAILS OF CONNECTOR J5

1.	GND	2.	GND
3.	NC	4.	NC
5.	NC	6.	GATE2
7.	OUT2	8.	CLK2
9.	OUT1	10.	GATE1
11.	CLK1	12.	GATE0
13.	OUT0	14.	CLK0
15.	NC	16.	NC
17.	NC	18.	NC
19.	NC	20.	NC
21.	NC	22.	$\overline{\text{PCLK}}$
23.	DRQ2	24.	DRQ1
25.	EXT1	26.	EXT2

7. DETAILS OF CONNECTOR J6

1.	NC	8.	EAR OUT
2.	NC	9.	GND
3.	(-Rx)	10.	NC
4.	(+Rx)	11.	(-12V)
5.	(-Tx)	12.	(+12V)
6.	(+Tx)	13.	Vcc
7.	(MIC IN)		

8. DETAILS OF CONNECTOR J7

1.	GND	14.	NC
2.	Tx	15.	NC
3.	Rx	16.	NC
4.	RTS	17.	NC
5.	CTS	18.	NC
6.	DSR	19.	NC
7.	GND	20.	DTR
8.	NC	21.	NC
9.	NC	22.	NC
10.	NC	23.	NC
11.	NC	24.	NC
12.	NC	25.	NC
13.	NC		

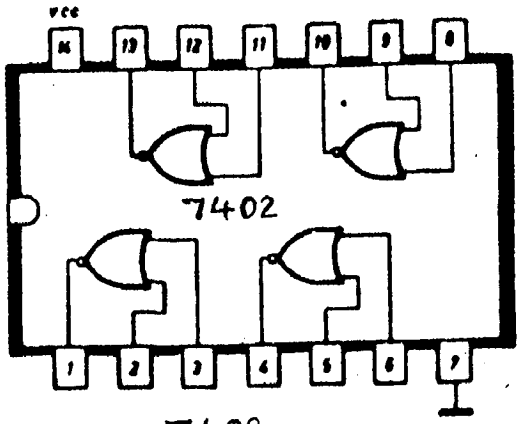
9) DETAILS OF CONNECTOR J8

1.	GND	14.	NC
2.	Tx	15.	NC
3.	Rx	16.	NC
4.	RTS	17.	NC
5.	CTS	18.	NC
6.	DSR	19.	NC
7.	GND	20.	DTR
8.	NC	21.	NC
9.	NC	22.	NC
10.	NC	23.	NC
11.	NC	24.	NC
12.	NC	25.	NC
13.	NC		

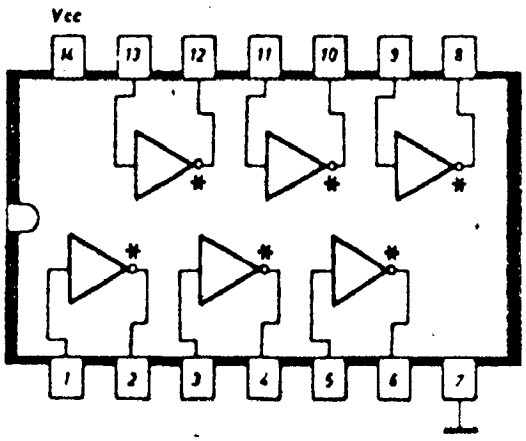
Rx	-	Receive Data
Tx	-	Transmit Data
CTS	-	Clear to send
RTS	-	Request to Send
DSR	-	Data Send Ready
DTR	-	Data Terminal Ready

APPENDIX C

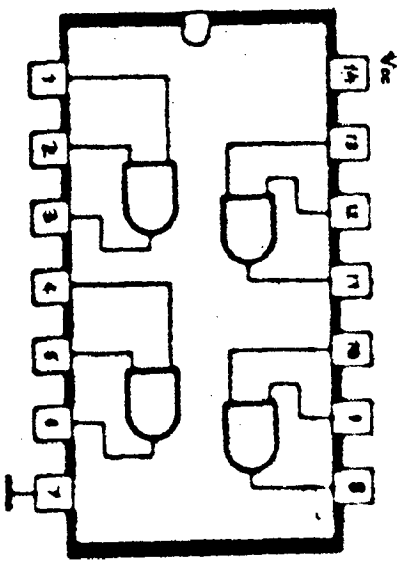
PIN DIAGRAMS OF THE I.C.s USED



7402

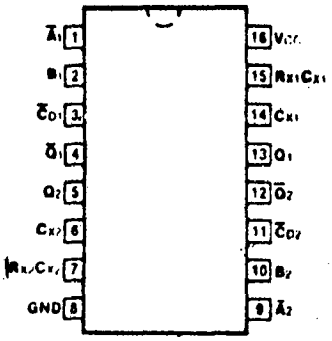


7404



7408

CONNECTION DIAGRAM
PINOUT

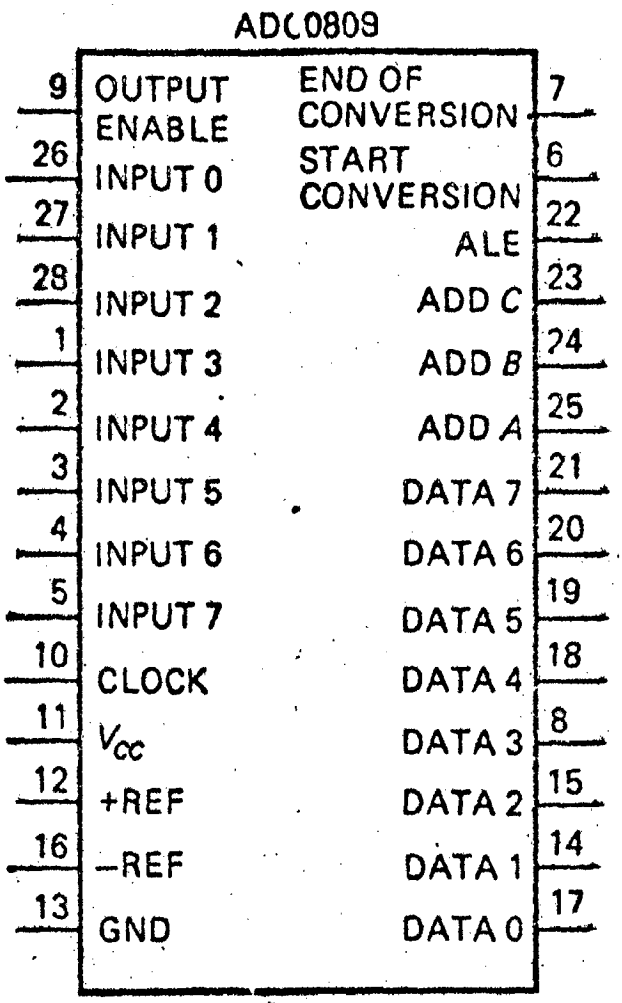


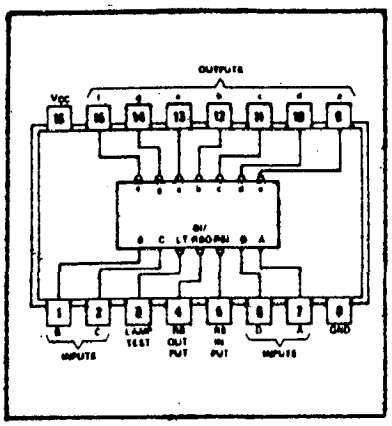
TRIGGERING
TRUTH TABLE

INPUTS			RESPONSE
A	B	$\overline{C}_D$	
X	X	L	No Trigger
↘	L	X	No Trigger
↘	H	H	Trigger
H	↗	X	No Trigger
L	↗	H	Trigger
L	H	↗	Trigger

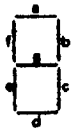
H = HIGH Voltage Level
L = LOW Voltage Level
X = Immaterial

74123

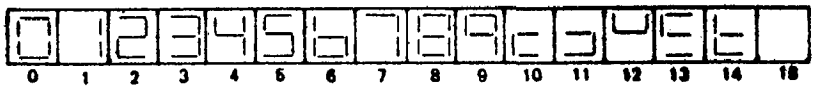

ADC0809
9
OUTPUT ENABLE
26
INPUT 0
27
INPUT 1
28
INPUT 2
1
INPUT 3
2
INPUT 4
3
INPUT 5
4
INPUT 6
5
INPUT 7
10
CLOCK
11
V_{CC}
12
+REF
16
−REF
13
GND
END OF CONVERSION
7
START CONVERSION
6
ALE
22
ADD C
23
ADD B
24
ADD A
25
DATA 7
21
DATA 6
20
DATA 5
19
DATA 4
18
DATA 3
8
DATA 2
15
DATA 1
14
DATA 0
17



(a)



(b) Segment identification

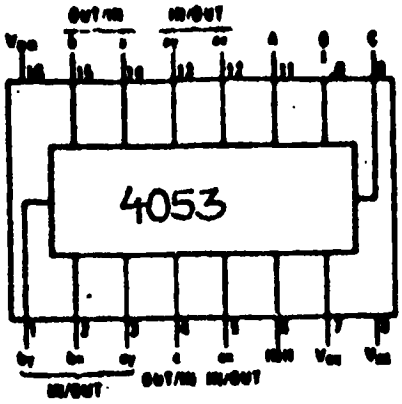


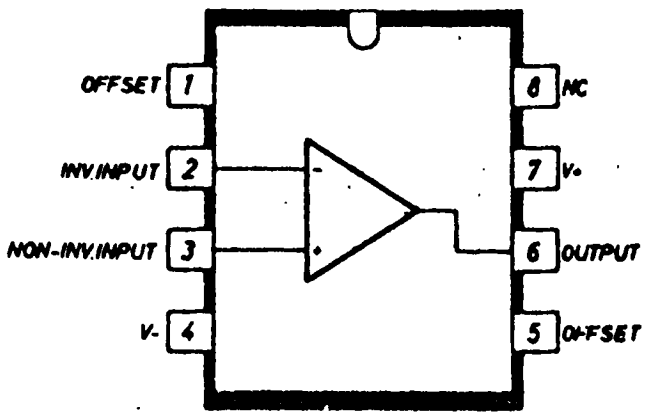
(c) Numerical designations and resultant displays

DECIMAL OR FUNCTION	INPUTS						BI/RBO†	OUTPUTS							NOTE
	LT	RBI	D	C	B	A		a	b	c	d	e	f	g	
0	H	H	L	L	L	L	H	ON	ON	ON	ON	ON	ON	OFF	1
1	H	X	L	L	L	H	H	OFF	ON	ON	OFF	OFF	OFF	OFF	1
2	H	X	L	L	H	L	H	ON	ON	OFF	ON	ON	OFF	ON	
3	H	X	L	L	H	H	H	ON	ON	ON	ON	OFF	OFF	ON	
4	H	X	L	H	L	L	H	OFF	ON	ON	OFF	OFF	ON	ON	
5	H	X	L	H	L	H	H	ON	OFF	ON	ON	OFF	ON	ON	
6	H	X	L	H	H	L	H	OFF	OFF	ON	ON	ON	ON	ON	
7	H	X	L	H	H	H	H	ON	ON	ON	OFF	OFF	OFF	OFF	
8	H	X	H	L	L	L	H	ON	ON	ON	ON	ON	ON	ON	
9	H	X	H	L	L	H	H	ON	ON	ON	OFF	OFF	ON	ON	
10	H	X	H	L	H	L	H	OFF	OFF	OFF	ON	ON	OFF	ON	
11	H	X	H	L	H	H	H	OFF	OFF	ON	ON	OFF	OFF	ON	
12	H	X	H	H	L	L	H	OFF	ON	OFF	OFF	OFF	ON	ON	
13	H	X	H	H	L	H	H	ON	OFF	OFF	ON	OFF	ON	ON	
14	H	X	H	H	H	L	H	OFF	OFF	OFF	ON	ON	ON	ON	
15	H	X	H	H	H	H	H	OFF	OFF	OFF	OFF	OFF	OFF	OFF	
BI	X	X	X	X	X	X	L	OFF	OFF	OFF	OFF	OFF	OFF	OFF	2
RBI	H	L	L	L	L	L	L	OFF	OFF	OFF	OFF	OFF	OFF	OFF	3
LT	L	X	X	X	X	X	H	ON	ON	ON	ON	ON	ON	ON	4

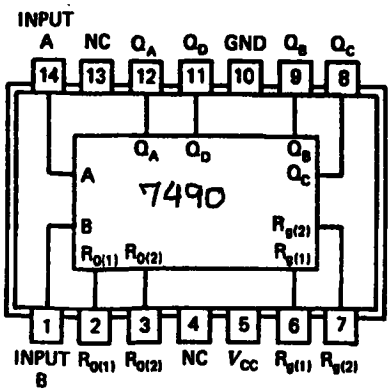
Function table

The 7447 display driver.


4053

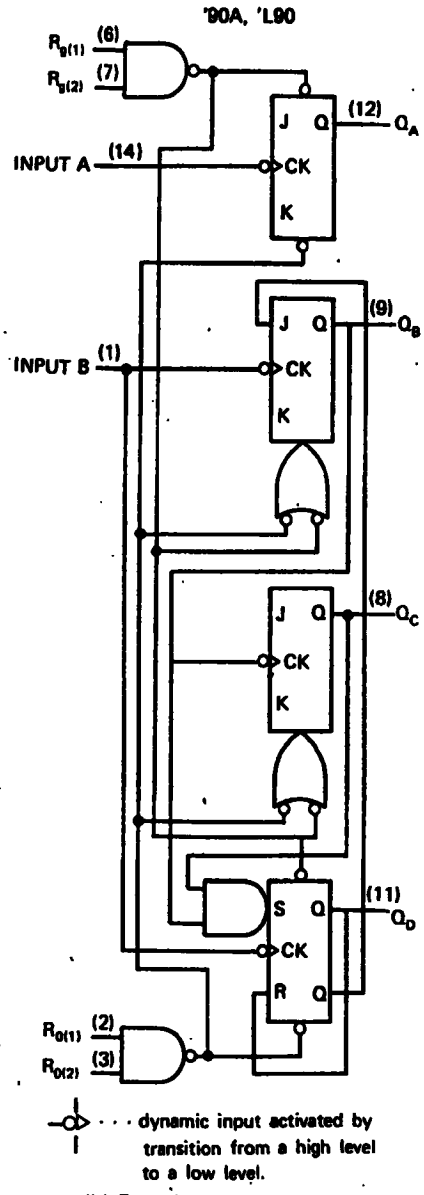

OFFSET
1
INV.INPUT
2
NON-INV.INPUT
3
V-
4
8
NC
7
V+
6
OUTPUT
5
OFFSET

741/3140



(a) Pin configuration

'90A, 'L90



(b) Functional block diagram

'90A, 'L90'
BCD COUNT SEQUENCE

COUNT	OUTPUT			
	Q_D	Q_C	Q_B	Q_A
0	L	L	L	L
1	L	L	L	H
2	L	L	H	L
3	L	L	H	H
4	L	H	L	L
5	L	H	L	H
6	L	H	H	L
7	L	H	H	H
8	H	L	L	L
9	H	L	L	H

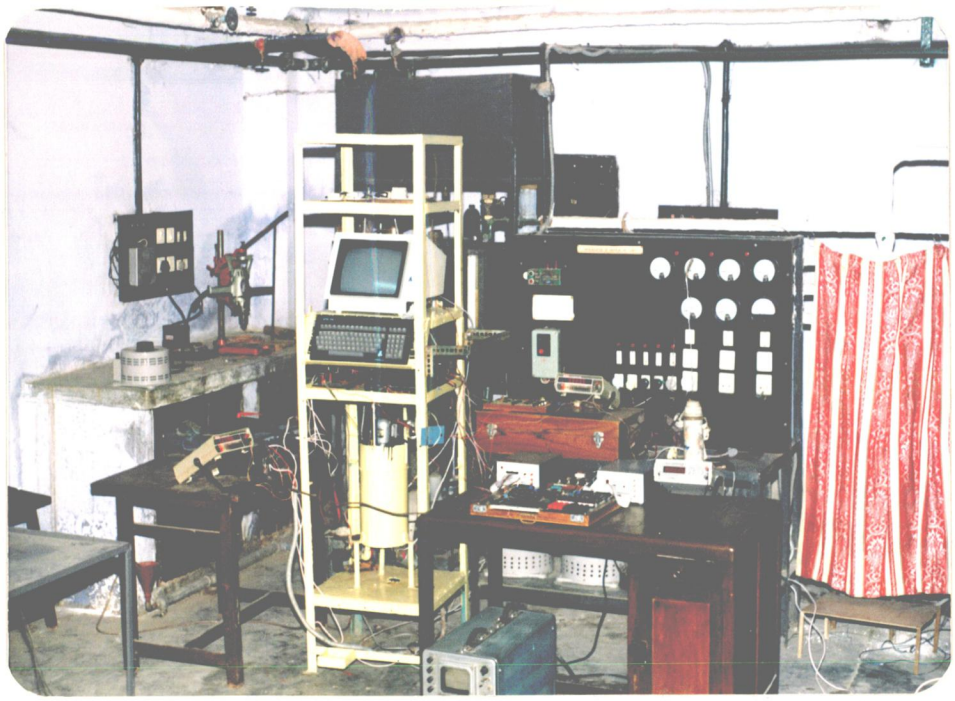
(c) Count sequence tables

'90A, 'L90'
RESET/COUNT FUNCTION TABLE

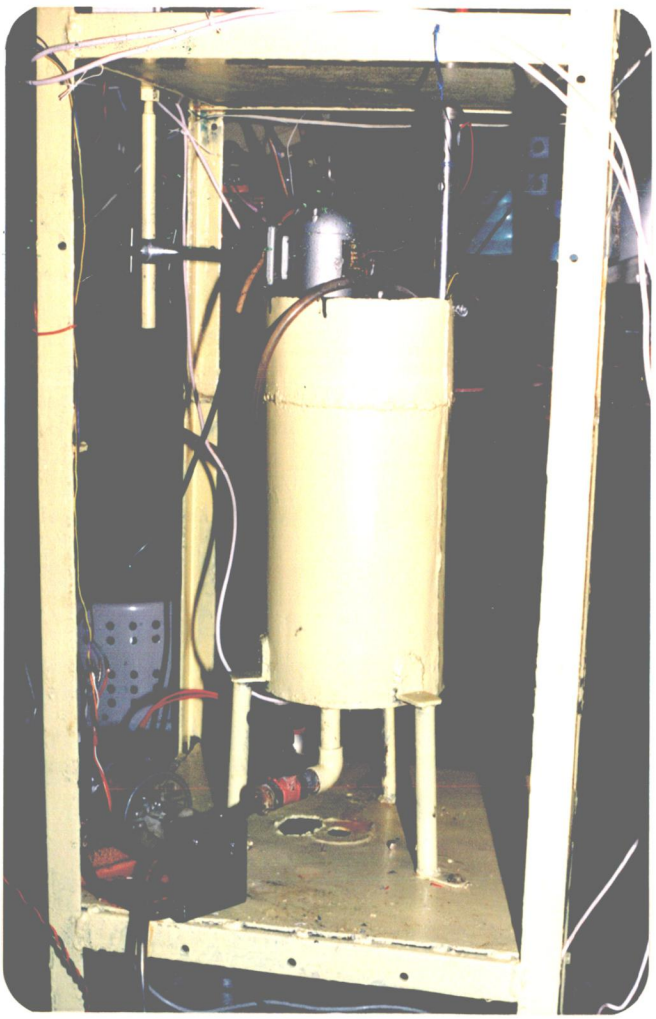
RESET INPUTS				OUTPUT			
$R_{0(1)}$	$R_{0(2)}$	$R_{9(1)}$	$R_{9(2)}$	Q_D	Q_C	Q_B	Q_A
H	H	L	X	L	L	L	L
H	H	X	L	L	L	L	L
X	X	H	H	H	L	L	H
X	L	X	L	COUNT			
L	X	L	X	COUNT			
L	X	X	L	COUNT			
X	L	L	X	COUNT			

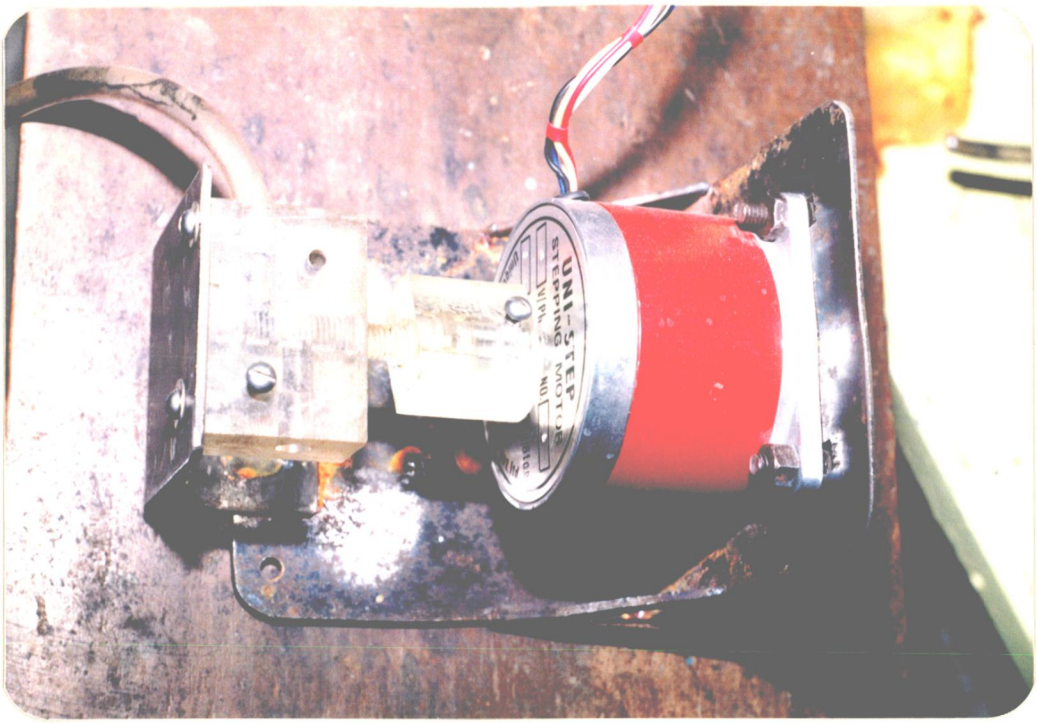
(d) RESET/COUNT function table

) 2) SYSTEM ASSEMBLY



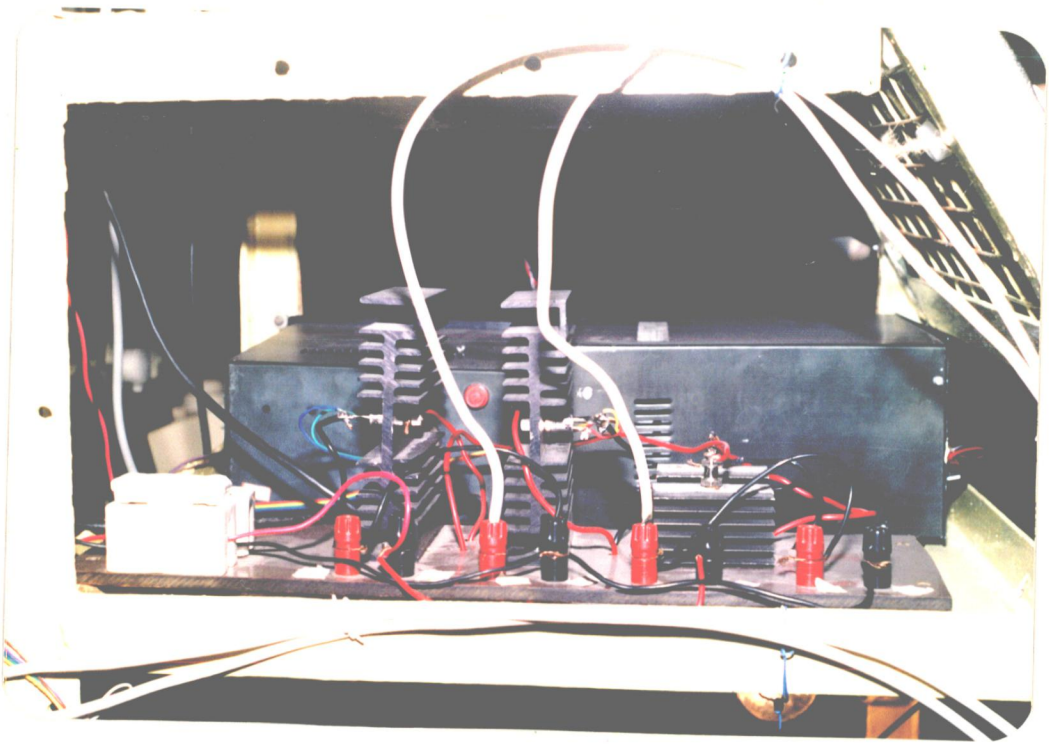
3) CSTR WITH STEPPER MOTOR CONTROLLED VALVE



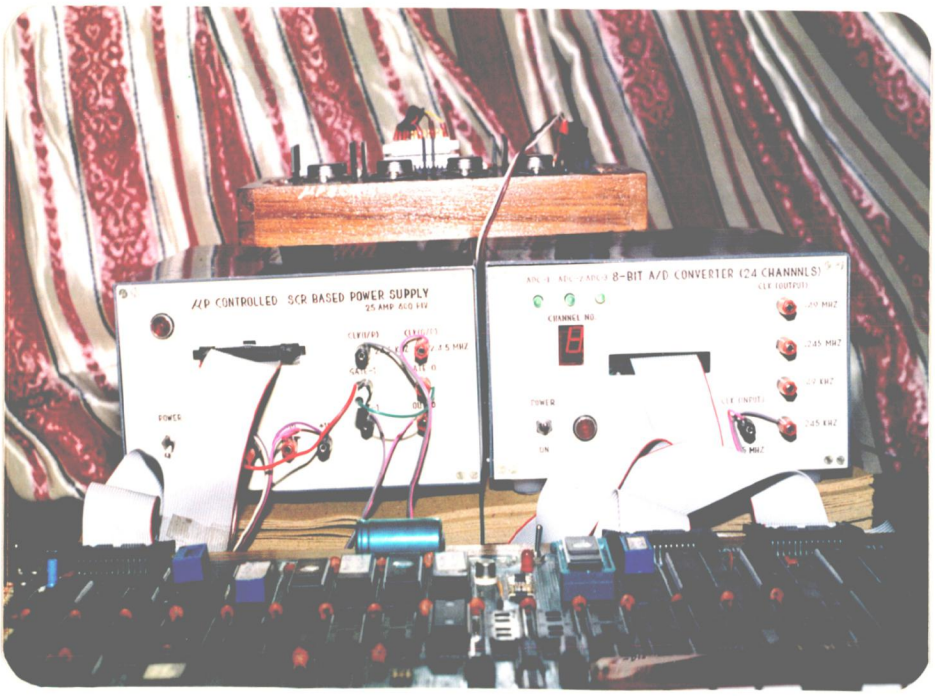


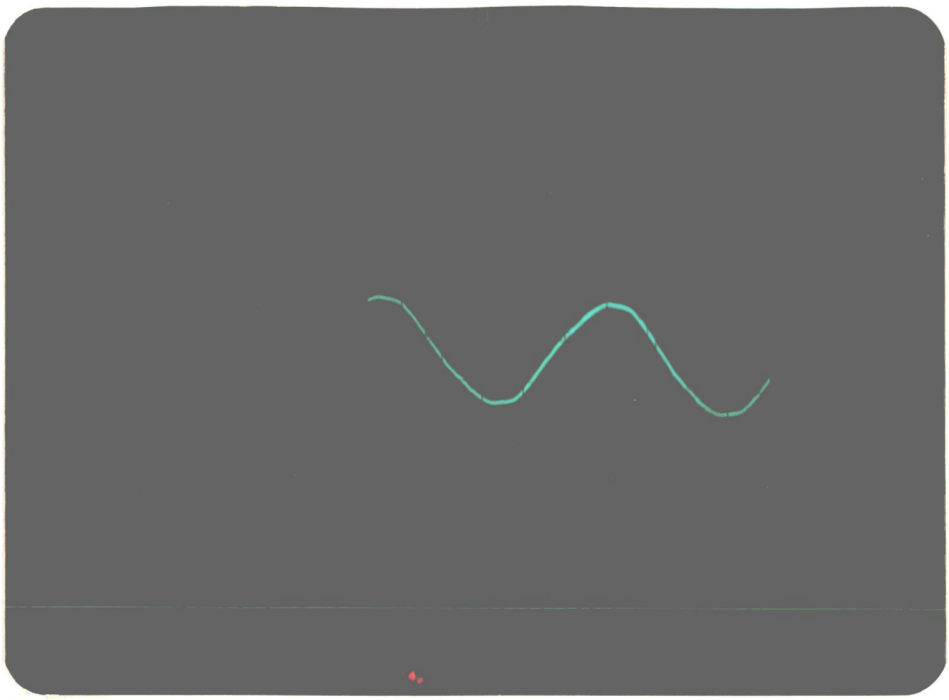
4) STEPPER MOTOR CONTROLLED VALVE

5) SCR CONVERTER BRIDGE

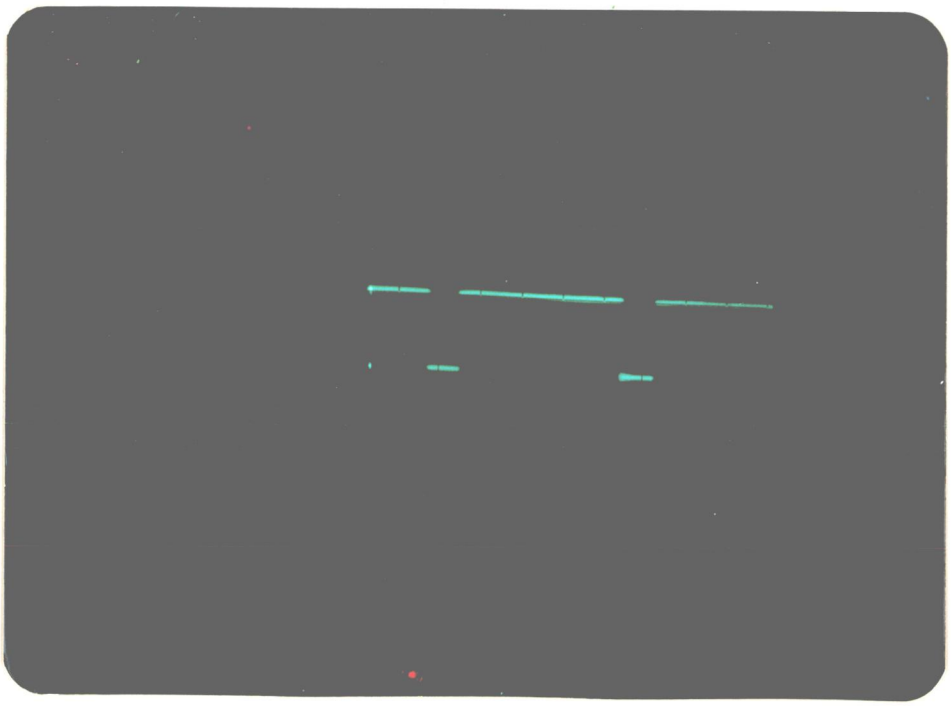


6-7. 8086 µp KIT WITH SCR FIRING CIRCUIT MODULE, ADC MODULE AND STEPPER MOTOR DRIVER CARD

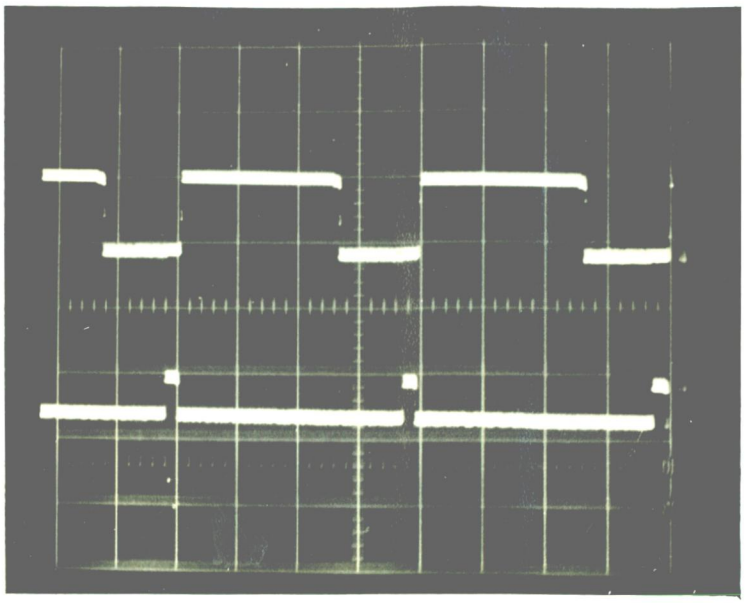




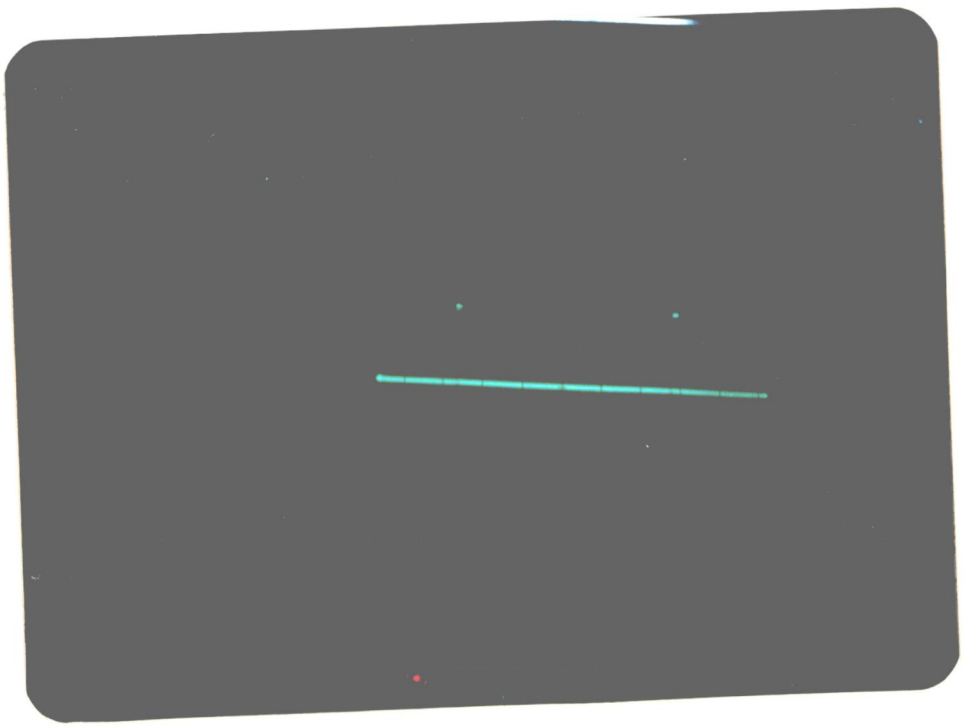
8. INPUT SINE WAVE TO CONVERTER BRIDGE



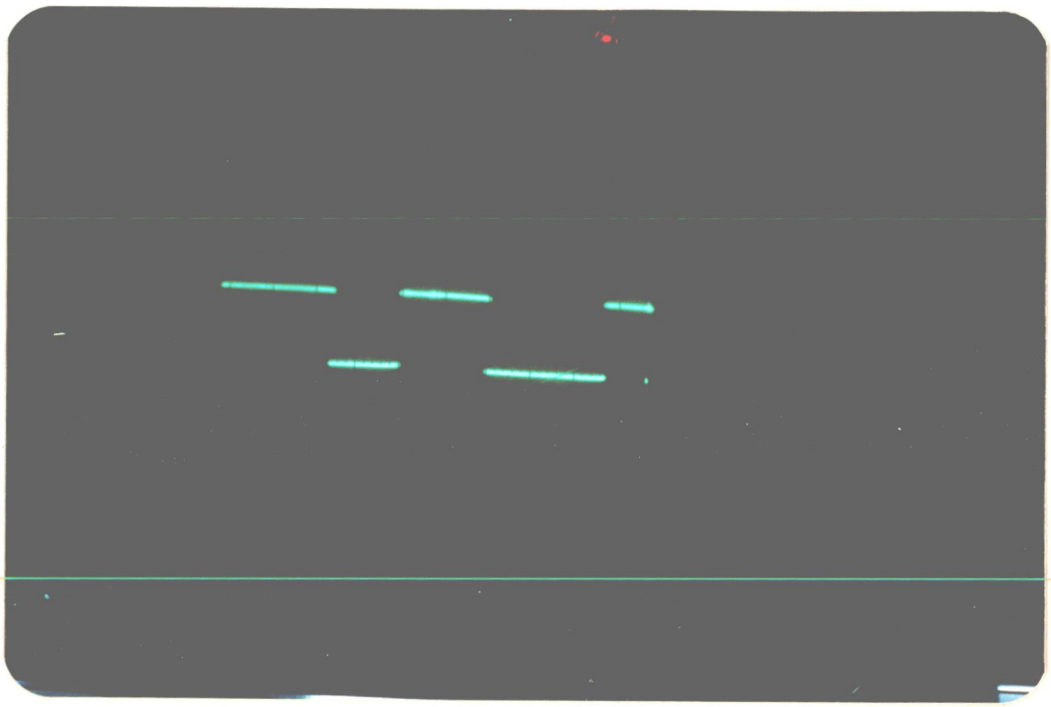
9. ZERO CROSSING O/P IN SCR FIRING CIRCUIT



10. TIMER O/P IN SCR FIRING CIRCUIT



11. FIRING PULSES TO SCRs



12. C-F CONVERTER O/P