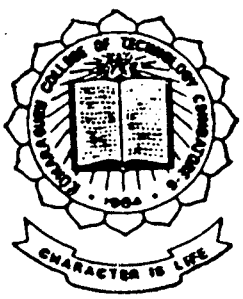
Micro Controller Based Timer/Counter/RPM Indicator

P-199

Project Work

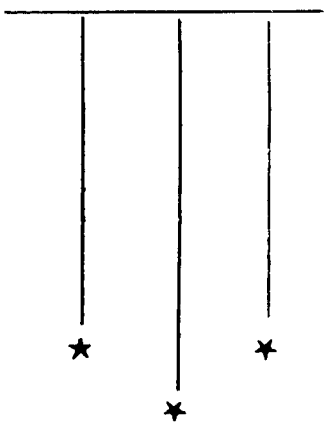


Submitted by

PARIMALAM L.

AJISH S.

ARUNACHALAM S.



Under the Guidance of

Dr. K. A. PALANISWAMY, Ph.D.,

IN PARTIAL FULFILMENT OF THE REQUIREMENTS
FOR THE AWARD OF THE DEGREE OF

BACHELOR OF ENGINEERING

IN ELECTRICAL AND ELECTRONICS ENGINEERING
OF THE BHARATHIAR UNIVERSITY, COIMBATORE

1993-94

Department of Electrical and Electronics Engineering

Kumaraguru College of Technology

Coimbatore - 641 006

Department of Electrical and Electronics Engineering

Kumaraguru College of Technology

Coimbatore - 641 006

CERTIFICATE

Name..... *Roll No.*.....

University Register No......

This is to certify that the Project work

MICRO CONTROLLER BASED TIMER/COUNTER/RPM INDICATOR

is a bonafide work carried out by

Mr. L. PARIMALAM & Miss. J. ARUNA

In partial fulfilment of the requirements for the award of the degree of Bachelor of Engineering in Electrical and Electronics Engineering Branch of the Bharathiar University Coimbatore during the academic year 1993–94

Station :

Date :

Dr. K. A. PALANISWAMY, M.E., M.Sc.(Engg.), Ph.D

Guide

Professor and Head

Department of Electrical and Electronics Engineering.

Kumaraguru College of Technology

Coimbatore 641 006

Dr. K. A. PALANISWAMY, M.E., M.Sc.(Engg.), Ph.D

H. O. D.

Professor and Head

Department of Electrical and Electronics Engineering.

Kumaraguru College of Technology

Coimbatore

Submitted for the University Examination held on.....

.....

Internal Examiner

.....

External Examiner

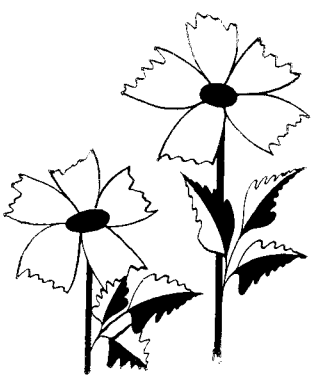
ACKNOWLEDGMENT

We express our deep sense of gratitude to Dr.K.A. PALANISWAMY, B.E., M.Sc.(ENGG.)., Ph.D., M.I.S.T.E., F.I.E., Professor and Head of the Department of Electrical and Electronics Engineering for his valuable guidance, encouragement, advice and suggestions in completing this project successfully.

We sincerely thank our Principal Dr.S. SUBRAMANIAM, Ph.D.,for providing the facilities to carry out the project in the college.

We also wish to convey our sincere thanks to all the members of staff of the Electrical and Electronics Engineering Department for their kind help in completing this project.

We also extend our sincere thanks to the faculty members of the Department of Electronics and communication Engineering for their whole-hearted co-operation by providing instruments for the testing purpose of this project.



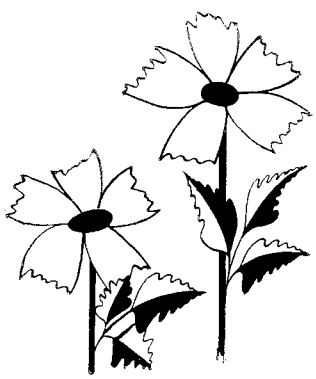
Synopsis

SYNOPSIS

In this project it is proposed to design and fabricate, a programmable timer/counter/rpm Indicator. A universal micro-controller 8048 is used as the processor which controls multi functions. The unit is very handy and can be used in a process industry for timing applications and counting.

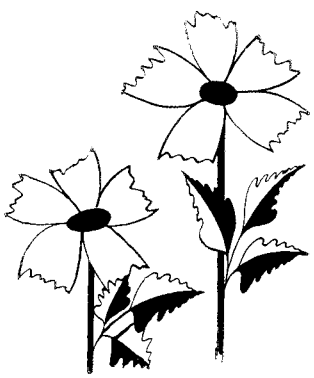
This project is more suitable for a cement industry with vapper motor control and timer for dedusting of electro-static precipitator plates in pollution control.

The test results are presented in this report.



Contents

CONTENTS



Introduction

CHAPTER I

INTRODUCTION

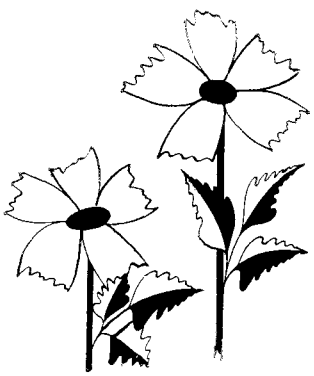
The heart of the process industry is the control applications. Any control application is based on a time slot. For example in a temperature control furnace the output of the controller goes active only after a desired time. Thus the timer places a very important role in a control processor.

Nowadays many industries adapt multioutput, PLC's (Programmable Logic Controller) which has also got a built in multioutput timers. But like a cement industry there may be many number of timers required which may be insufficient and very far from the control room. Thus a purely isolated multioutput timer is designed in one project.

There may be only one output with a particular time in an ordinary timer. But using the controller will provide a program based unit i.e., the user can have in any option without increase the cost of course with his brain.

The project also includes an even counter which may be up and down which is also used in the level finding in process industry. Counters are used in a wide variety of counting application in scientific instruments, industrial controls, computers and communication equipments. The basic function of counter is to 'memorize' how many clock pulses have been applied to the input; hence in the most basic sense counters are memory systems. They are used for counting pulses, equipment operation sequencing, frequency division and mathematical manipulation. Because of the wide range of application for counters, as well as their large demand, manufacturers offer a wide selection of off-the-shelf MSI counter circuits, differing in complexity, functional, versatility, speed, power and cost. This type of counter can also be made to count in either the forward or reverse direction.

In addition to the above features this unit can also indicate the rpm of a given motor or of any rotating device.



System Block Diagram

CHAPTER II

SYSTEM BLOCK DIAGRAM

Fig. 2.1 gives a block diagram of the microcontroller based Timer / counter / rpm Indicator. The various unity shown in the block diagram are described in this chapter.

2.1 CONTROLLER

The controller is the heart of the project . An 8 bit controller is found suitable and is used for our project. The EPROM version of the 8048 family and saves a lot of 8748. The controller controls external hardware of all other associatries. Like key / display and Input - Output structures. This controller has a built in CPU, 64 bytes of RAM and IKB of EPROM.

2.2 DISPLAY / KEY STRUCTURE

This gives the visual indication of what the unit does. The display unit is a common anode 4 digit seven segment unit which is driven by transistors. The display board has five switches which are used for the user operation. Like programming stopping and starting.

2.3 OUTPUT SECTION

The output section is the periphery of the unit. This consists of driving transistors and indicating LED's. The glowing of the LED indicates that the output is active.

These transistor / LED are driver by a buffer. The Buffer is a protective device used in interfacing. The output of the output board are connected to the target that is the device to be controlled.

2.4 SENSOR

For the counting operation and the rpm indication, the signals are obtained by the defecting devices called as sensors. A sensor is a transducer. A proximity sensor is used in our project.

CHAPTER-1

SYSTEM BLOCK DIAGRAM

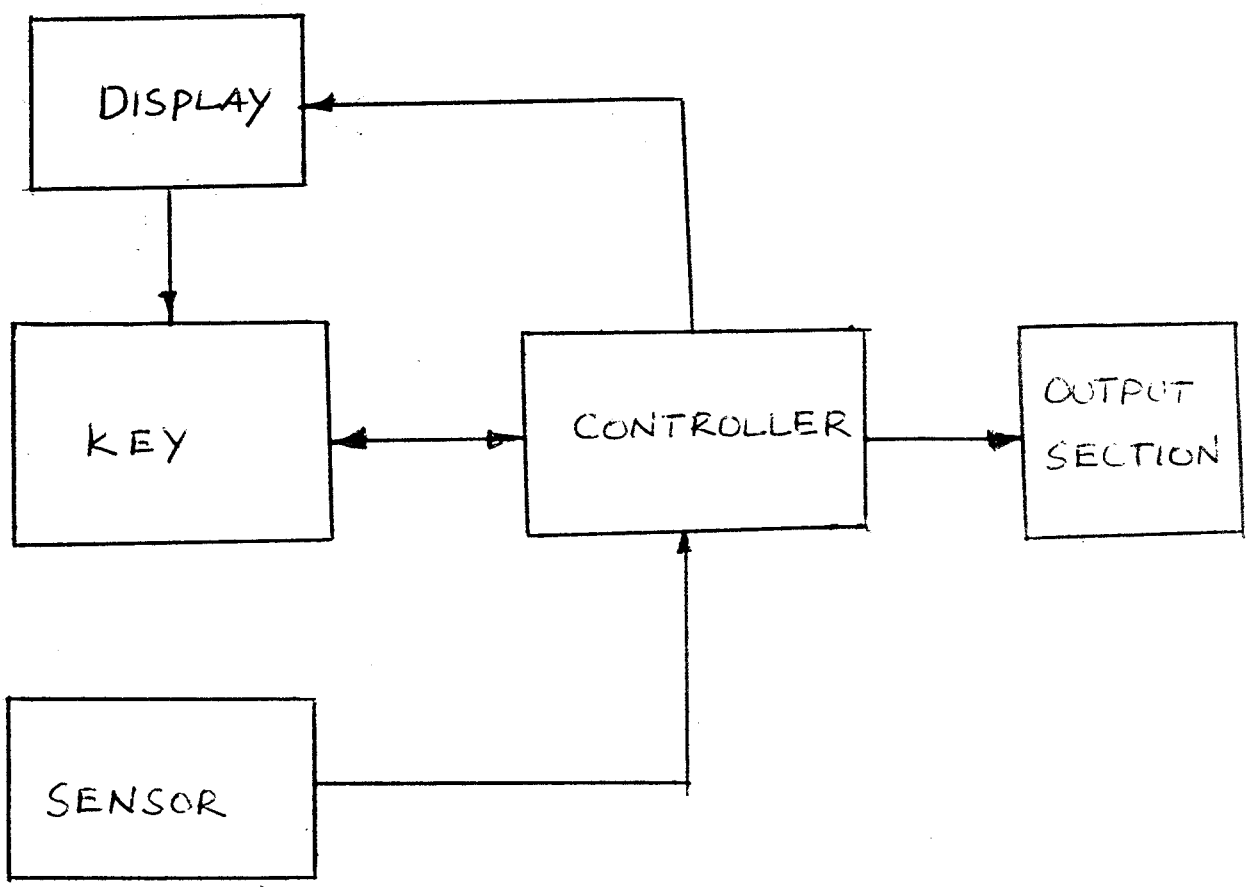
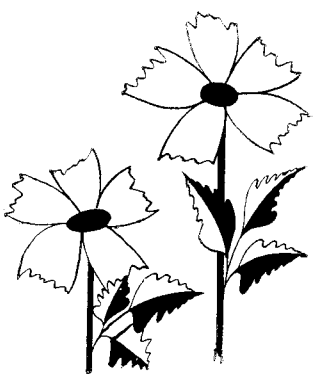


Fig-2.1



System Hardware and Design

CHAPTER III

DESIGN OF SYSTEM HARDWARE

3.1 INTRODUCTION

The detailed block diagram of the micro controller is shown in Fig. A.1.3. The design of system Hardware using the microcontroller is discussed in this chapter.

The SS (Pin 5), Vcc (Pin 26) and Vdd (Pin 40) are connected to the +5v DC supply. Pin 20 is connected to ground. The system controller clock is generated by means of a 6 MHz crystal connected between pin 2 (XTAL1) and pin 3 (XTAL2).

Port 2 pins 2.0 to 2.4 are used for display and port 1 pins 1.5 to 1.7 are connected to the fact keys which is used when various modes are selected.

3.2 DISPLAY UNIT

The type of display used in this timer / counter / rpm indicator is a common anode seven-segment LED display. The display unit has four digits.

Port 1.0 pin is connected to the least significant digit while port 1.4 is connected to the fifth key and 1.0, 1.2, 1.3, 1.4 are connected to the corresponding keys.

All the anodes of the four seven segment LED's are tied to a 5V supply through PNP transistors (2N2907A) which acts as a switch.

3.3 SYSTEM HARDWARE

The system hardware can be divided into the following categories.

1. CONTROLLER SECTION
2. DISPLAY SECTION
3. KEYS SECTION
4. OUTPUT SECTION
5. SENSOR INPUT SECTION

3.3.1 CONTROLLER SECTION

The heart of the hardware is the controller part. The internal structure of 8748 is described in Appendix A. In this the bi-directional bus DB0 to DB3 is used for selecting the digits for the display. Proper data on this bus will select or switch on the particular transistor to enable the common anode of the display.

The port 1 P10 to P17 are used as outputs. These will send the seven segment code to the display. The port pins P1.0 to P1.4 are used as inputs to sense the status of the keys. Since P1.0 to P1.4 act both as input and output, there should be a latch to separate them avoiding both getting clashed i.e., P1.0 to P1.4 should not act as input and output at a time.

The RD signal is used to strobe the latch. DB4 pin is used to enable the keys. A decoupling capacitor is used to filter the noise between the VCC and ground pin of the processor. The reset pin is provided with power on reset facility. T1 pin is used for counting the pulses. A crystal serves as the time base.

As the interrupt is not needed it is pulled high.

3.3.2 DISPLAY SECTION

A four digit common anode seven segment display is used. The common anode of each digit is connected to the output of transistor. The action of display is in the multiplexed form. The seven segments are derived from the outputs of the latch. For each segment a current limiting resistor of 100 ohms is provided.

3.3.3 KEYS SECTION

There are five keys used in this project. The closure of a key is sensed by the port pins P1.0 to P1.4. Each key has a defined function. The keys provided are of push button type. The keys used here are tact keys which are widely used for electronic devices. The key structures are programmed as follows

K1 Key

PROGRAM MODE	NEXT PARAMETER	UP	DOWN	RESET
RUN MODE	NEXT MODE	START	STOP	CONTINUE

The key K1 is used, to toggle between program mode / run mode. The functions of keys K1, K2, K3, K4 and K5 are described below.

Key	Program mode	Run mode
K2	It jumps to ask the next parameter.	It jumps to the next mode.
K3	It just increments the value	It starts the unit
K4	It just decrement the value	It stops the unit
K5	It resets the process	It continues from the stop position.

3.3.4 OUTPUT SECTION

In this section the transistors drives the target devices according to the data given to its input. The NPN transistors are used for this purpose. The input to the transistor are given from port 2 of microcontroller through base current limiting resistors. The output indication is through the light emitting diodes.

3.3.5 SENSOR INPUT SECTION

During the counting and rpm indication the pulses from the sensor has to be filtered from noise. For this an RC filter is used. The zener diode will act as a high spike suppressing device.

3.4 DESIGN OF HARDWARE

3.4.1 DESIGN OF RESISTANCES FOR THE SEGMENTS R01 TO R08.

Maximum current for each segment is 20mA. Since the output voltage Vg is 5V,

$$R = \frac{5V}{60mA} = 300 \text{ ohms}$$

Therefore a value of 300 ohms is chosen for the segments R01 to R08.

3.4.2 SELECTION OF BASE RESISTANCE

The required value of the base resistance Rb is given by

$$IB = \frac{VCC - VBE}{Rb} \qquad ----- \quad (3.1)$$

Since the voltage VBE across the forward biased junction is approximately 0.2V for a germanium transistor and 0.6V for a silicon transistor. The equation (3.1) is approximated as

$$IB = VCC / Rb \qquad ----- \quad (3.2)$$

The selected base current is 1.85 mA. So Rb is approximately 2.7 Kilo ohms.

3.4.3 DESIGN OF RC FILTER

The noise is maximum usually in an automobile during spark-ignition. Taking this worst condition, the pulse width of the noise will not be greater than 50 to 80 microseconds.

Therefore, if R = 10K

C = 0.1 micro Farad

RC = 10 x 0.1 x 10 x 10

= 1 millisecond

Then RC >> 80 microseconds. So select R = 10K and C = 0.1 micro Farad.

3.4.4 SELECTION R20 - R28

The collector current IC flowing through the resistor RC is given by

$$IC = \frac{-\ VCC - VCE\ (sat)}{RC} \qquad - - - - - \quad (3.3)$$

Neglecting VCE(sat) [since VCE(sat) <<< VCC]

The collector current IC = VCC / RC - - - - - - (3.4)

Taking the maximum collector current IC = 10 mA,

RC = VCC / IC = 5 / (10 x 10) = 500 ohms

so 1K is selected for RC for better protection.

3.5 WORKING OF THE HARDWARE

The working of the hardware can be divided into the following categories.

1. Working of CPU
2. Working of keys
3. Working of display
4. Working of input section
5. Working of output section

3.5.1 WORKING OF CPU

The CPU has the main role in working of the unit. The working of the unit is completely controlled by the software. DB0 to DB3 scans the display transistor so as to multiplex the digits. The seven segment data which is to be displayed on the digits are latched to the port 1 with the help of an 8 bit latch 74HC373. To latch, a strobe is necessary, by using a proper command a RD signal is generated. DB4 is made low whenever it has to be disabled.

Depending upon the mode of operation, the data are put out through port 2 to the output board.

3.5.2 WORKING OF KEYS

The common end of keys are connected to the enable pin of the CPU i.e., DB4. When DB4 is in mode 0, if any key is closed the particular diode will be forward biased and it conducts and the particular port pin goes low. By taking this key data inside the key proper action is taken.

3.5.3 WORKING OF THE DISPLAY

The display unit is multiplexed. The scanned outputs from DB0 to DB3 will select the transistor in a sequence.

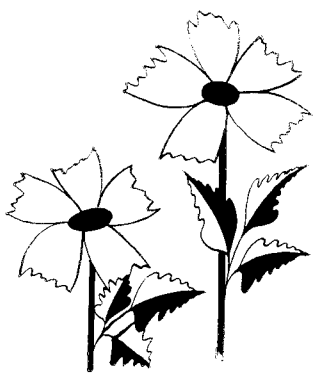
This results in selecting the digits in sequence. First the digit is selected and then the seven segment data is latched into the latch. Thus, we get the number of the digit displayed. The RD is generated by executing any MOV X instruction. The digit selecting transistors are PNP transistor. Here we use 2N2907A transistors.

3.5.4 WORKING OF THE OUTPUT SECTION

Whenever a high signal is given to the base of the NPN transistor, the transistor goes to the ON state and the LED glows. Instead of a LED, if a relay is given, the relay gets energized and the output is activated. Depending on the mode of operation the outputs are made to energize.

3.5.5 WORKING OF THE INPUT SECTION

Whenever a magnetic substance comes very near to the proximity switch, the proximity switch will produce a pulse. That pulse is filtered and shaped into a pulse by using a schmitt trigger or a transistor and then it is given to the T1 input of the controller. The processor then accepts this pulse and then counts with the help of the software.



Modes of operation

CHAPTER IV

MODES OF OPERATION

The following are the modes of operation of this unit.

4.1 MODE 0

MODE 0 is single cycle mode i.e., the operation of output is confined only for one cycle. Here the output is energized in a sequence starting from the zeroth output tc the last output. The number of outputs are 8. In this mode whenever one output is enabled the previous is cleared. The ON time of the output is equal to the time delay between two outputs.

Thus we can say the output will be energized after 8 times the delay between each output.

In this mode once the cycle is overall the outputs are disabled. After the mode is selected the start key has to be pressed. Once the complete cycle is over the cycle can be repeated again by pressing the continue key.

If the output has to be disabled in between the cycle, stop key can used and continue key to continue operation.

4.2 MODE 1

MODE 1 is a multicycle mode. The operation of this is also similar to that of MODE 0. Once the MODE 1 is selected, it waits for the start key. After pressing the start key the operation starts. During every enabling output, the previous output is cleared. But after finishing one cycle, it again repeats automatically. Here also stop and continue can be used to interrupt the operation.

4.3 MODE 2

MODE 2 is resumed output mode. In this mode of operation the cycle is repeated as per MODE 1. As the start key is pressed the outputs energize in a sequence. But the previous output DOES NOT CLEAR i.e., a Johnson counter action takes place. Stop and continue operations are similar as in other modes.

4.4 MODE 3

MODE 3 is all similar mode. Sometime the same timing has to be provided for many outputs more than 10 or 12. Thus one output cannot be used to all the loads due to port loading problem. Hence all the outputs are made high at a time and low at a time. The rest of the operations are similar to the other modes.

4.5 MODE 4

MODE 4 is manual mode. This mode is also called pulsating mode. Whenever the start key is pressed the first output gets enabled. The second output can be made enable only when start key is again pressed manually. The previous output will be cleared after the next output gets enabled.

4.6 MODE 5

MODE 5 is a COUNTER mode. Unlike the above modes this is purely an input based mode i.e., the operation is dependent on the input to the controller. Here it counts the number of the pulses given to the input. The counting is in the upward fashion. The maximum count is 9999. Once the maximum count is reached, the counter gets reset itself.

4.7 MODE 6

MODE 6 is a DOWN counter mode. This mode is also an input dependent mode like MODE 5. The operation is dependent on the input to the controller. Here it counts the number of pulses given to the inputs. The counting is in the downward fashion. Once the least count 0000 is reached the counter gets reset itself.

4.8 MODE 7

MODE 7 is a up divider mode. In this mode the pulses are counted in the similar fashion to the MODE 5. But the counter increments only after the pre-scalar value is reached i.e., every increment is nothing but the number pulses divided by the pre-scalar. The value of pre-scalar is programmable. The maximum value of the pre-scalar can be 99. But this can be extended by 9999 by slight change in software.

In this mode the maximum value to be counted will be the highest multiple of the pre-scalar i.e., lesser cr equal to 9999.

4.9 MODE 8

MODE 8 is a down divider mode. In this mode the pulses are counted in the similar fashion as in the MODE 6. But the counter decrement only after the pre-scalar value is reached i.e., the every decrement is nothing but the number of pulses/the pre-scalar. The value of the pre-scalar is programmable. The minimum value of the pre-scalar is 00.

In this mode, the minimum value to be counted will be the lowest multiple of the pre-scalar i.e., higher or equal to 0.

4.10 MODE 9

MODE 9 is the rpm indication mode. In this mode the rpm of any rotating device is displayed. The time response and the resolution depends on the number of teeth on the rotating shaft. For example, if there is only one tooth to get a resolution of one rpm, the time response will be very slow i.e., of one minute. But if there are 10 teeth the time response for the same resolution will be 10 seconds. Thus by increasing the number of teeth we can have better time response with good resolution.

Microcontroller **8748**

CHAPTER V

MICROCONTROLLER 8748

5.1 INTRODUCTION

The heart of the energy meter is the system controllers. In order to provide room for flexibility and versatility, the use of such a system controller becomes essential. Instead of opting for a microprocessor which requires the use of additional peripheral devices like program memory, data memory, 8255 PPI etc, choice of a microcontroller is more suitable.

Microcontrollers are microcomputers used for dedicated applications incorporating clock, CPU, RAM, ROM, I/O ports and interrupt capability all within a single chip. Hence, the use of microcontrollers increases the efficiency, reduces overall cost, occupies less space and is more advantageous. Some of the benefits which arise from having a one-chip microcomputer are:

1. Small size and power for the controller portion of an instrument.

2. The opportunity to identify one chip as a "kernel" for the digital portion of an instrument for self-test purpose.

3. The definition of an especially efficient instrument set, with mostly one byte instrument.

5.2 SELECTION OF PROCESSOR

Selection of a microcontroller well suited to our application requires a wide range of analysis of the existing microcontrollers. Intel 8048 and 8051 series, Motorolas M6801 series, the MOS Technology 370, Texas Instruments TMS 1000 Zilong's Z8 and Toshiba's OKI series are some of the microcontrollers available.

Most of these microcomputers are 8-bit microcomputers except TMS 1000 which has a word length of 4 bits. The Intel 8748, which is also very popular is a member of Intel 8048 series. The Intel 8051 is the latest single chip 8-bit microcomputers which has a very powerful instructicn set and operates with 12MHz clock. Intel 8096 series of microcomputer are 16-bit single chip microcomputers. Of these C's available, the final bid was made on 8748. This microcontroller has a program memory of 1k bytes and a data memory of 64 bytes. It has 2 ports of 8 pins each and a bi-directional databus (8 pins). It has an inbuild timer/counter also. All these options suit our requiremments very well. In terms of cost also 8748 is the most economical processor of all other controllers and hence chosen.

5.3 ARCHITECTURE OF 8748

Intel single chip C 8748 is pin compatible with 8048 which is considered to be the head of Intel's-MCS-48 family of UCs. The instruction set for both of them are same.

8748 is provided with a 8 bit CPU, 1K * 8 ROM program memory, 64 * 8 RAM data memory, 27 I/O lines and an 8-bit timer/event counter. The main advantage is that either the capacity of program memory of data memory or even both can be expanded by connecting memory chips externally.

ARCHITECTURE

ARITHMETIC SECTION

The arithmetic section of the processor contains the basic data manipulation function of the 8748 and can be divided into the following blocks:

1. Arithmetic Logic unit
2. Accumulator
3. Carry flag
4. Instruction decoder.

INSTRUCTION DECODER

The operation code portion of each program instruction is stored in the instruction decoder and converted to outputs which control the function of each of the blocks of the Arithmetic section. There lines control the source of data and the destination register as well as the function performed in the ALU.

ALU

The ALU accepts 8-bit data words from one or two sources and generally an 8-bit result under control of the instruction decoder. The ALU can perform functions like Add with or without carry, AND, OR, EX-OR, Increment / Decrement, Bit complement, Rotate left, right, swap nibbles BCD decimal adjust etc. If the operation performed by the ALU results is a value represented by more than 8-bits, a carry flag is set in the program status word.

ACCUMULATOR

Accumulator is the single most important data register in the processor, being one of the sources of input to the ALU and often the destination of the result of operations performed in the ALU. Data to and from I/O ports and memory also normally passes through the accumulator.

PROGRAM MEMORY

Resident program memory consists of 1024, 2048 & 4096 words eight bits wide which are addressed by the program counter. In the 8748 this memory is user programmable and erasable EPROM. There are three locations in program memory of special importance. They are:

1. Location 0: Activating the reset line of the processor causes the first instruction to be fetched from location 0.

2. Location 3: Activating the interrupt input line of the processor causes a jump to subroutine at location3.

3. Location 7: A timer/ counter interrupt resulting from timer / counter overflow causes a jump to subroutine at location 7.

Therefore the first instruction to be executed after initialization is stored is location 0, the first word of the interrupt service subroutine is stored in location 3 and the first word of a timer / counter service routine is stored in location 7.

Program memory can also be used to store constants as well as program instructions. The program memory map has been shown in Appendix B.

DATA MEMORY

Resident data memory is organized as 64 words 8-bit wide. All locations are indirectly addressable through either of RAM the pointer registers which reside at address 0 and 1 of the register array. In addition as shown in the diagram the first eight locations of the register array are designated as the working register and are directly addressable by several instructions. Since these registers are easily addressed they are used mostly to store more frequently accessed intermediate results.

By executing a register bank switch instruction (SELRB) loc 24-31 over designated as working registers in place of loc 0-7 and are then directly addressable. This second bank of registers may be used as an extension of the first bank or unserved for use during interrupt service subroutines allowing the registers of bank or unserved for use during interrupt service subroutines allowing the registers of bank 0 used in the main program to be instantly 'saved' by a bank switch. Registers R0 & R1 are a part of the working register array, bank switching effectively creates two more pointer registers which along with R0 &R1 can be effectively used to access up to four separate working areas in RAM at a time.

RAM locations (8-23) also serve a dual role in that they contain the program counter stack. There locations are addressed by the stack pointer as well as by the pointers R0 and R1. The data memory map has been shown in Appendix B.

INPUT / OUTPUT

It has got 27 lines which can be used for input or output functions. These lines are grouped as 3 ports of 8 lines. These serve as either I/P< O/P or bi-directional ports and 3 test I/Ps which can alter the program sequences when listed by conditional jump instruction.

Ports 1 and 2 are each 8-bit wide and have identical characteristics. As input ports these are non-latching i.e., inputs must be present until read by an input instruction. The lines of port 1 and 2 are called quasi-directional because of a special output circuit structures which allows each lines to serve as an I/P< O/P or both, even though O/Ps are statically latched. Each lines is continuously pulled up to Vcc through a resistive device of relatively high impedance.

It is important to note that the ORL and ANL are read / write operations. When executed, the C "reads" the port, modifies the data according to the instructions, then

"writes" the data back into the port. The "writing" enables the low impedance pull-up momentarily again even if the data was unchanged from a "1". This specifically applies to configuration that have inputs and 1 outputs mixed together on the same port.

BUS

Bus is also an 8-bit port which is a true bi-directional port with associated input and output strobes. If the bi-directional feature is not needed, bus can serve as either a statically lathed output port or non-latching input port. I/P and O/P lines on this port cannot be mixed however.

As static port, data is written and latched using the OUTL instruction and inputted using the INS inst. The INS and OUTL unit generate pulses on the corresponding RD and WR output strobe lines, however, in the static port mode they are generally not used. As a bi-directional port the MOVX instructions are used to read and write port. A write to the port generates a pulse on the WR output line and output data generates a pulse on the RD output line and input data must be valid at the trailing edge of RD. When not being written or read, the BUS lines are in a high impedance state.

TEST AND INT. INPUTS

Three pins serve as inputs and are testable with the conditional jump inst. These are T0, Tl AND INT. These pins allow inputs to cause program branches without the necessity to load an input port into the accumulator. The T0, Tl and INT pins have other possible functions as well.

PROGRAM COUNTER AND STACK

The program counter is an independent counter and the program counter stack is implemented using pairs of registers in the data memory array. The program counter is initialized to zero by activating the RESET line.

An interrupt or CALL to a subroutine causes the contents of the program counter to be stored in one of the 8-register pairs of the program counter stack as shown.

The pair to be used is determined by a 3 bit stack pointer which is part of the program status word (PSW) data RAM locations 8-23 are available as stack registers and are used to store the current program counter value and 4 bits of PSW as shown.

Nesting of subroutines within subroutines can continue up to eight times without over flowing the stack. The end of a subroutine, which is signaled by a return instruction causes the stack pointer to be decremented and the contents of the resulting register pair to be transferred to the program counter.

PROGRAM STATUS WORD

An 8-bit word which can be loaded to and from the accumulator exists called the program status word. The PSW is a collection of flip-flop which can be read or written as a whole. The ability to write to PSW allows for easy restoration of machine status after a power down sequence.

The upper four bits of PSW are stored in the program counter stack with every call to subroutine or interrupt vector and are optionally restored upon returns with the RETR instruction. The RET instruction does not update PSW.

PSW:

Bits 0-2 - Stack pointer bits (S0, S1, S2)

Bit 3 - Not used ('1' level when read 1)

Bit 4 - Working register back switch bit (BS)

0 = Bank 0

1 = Bank 1

Bit 5 - Flago bit (FO) user controlled flag which can be complemented or cleared and tested with the conditional jump instruction JFO.

Bit 6 - Auxillary carry (AC) bit generated by an ADD instruction and used by the decimal adjust instruction DAA.

Bit 7 - Carry (CY) carry flag which indicates that the previous operation has resulted in overflow of the accumulator.

CONDITIONAL BRANCH LOGIC

The conditional branch logic within the processor enables several conditional branches internal and external to the processor to be tested by the users program.

INTERRUPT

An interrupt sequence is initiated by applying a low '0' level input to the INT pin. The interrupt line is sampled every instruction cycle and when detected causes a "call to subroutine" at location 3 in program memory as soon as all cycles of the current instruction are complete. For 2 cycle instruction the interrupt line is sampled in the 2nd cycle only. The INT must be held low for atleast 3 machine cycles to ensure proper interrupt operations. As in any CALL to subroutines, the program counter and program status word are

saved in the stack. The interrupt system is single level in that once an interrupt it detected all further interrupt requests are ignored until execution of an RETR re-enables the interrupt input logic. This occurs at the beginning of the second cycle of the RETR instruction. This sequence holds true also for an internal interrupt generated by timer, overflow. If an internal timer / counter generated interrupt and an external interrupt are detected at the same time, the external source will be recognized.

TIMER / COUNTER

The 8048 contains a counter to aid the user in counting external events and generating accurate time delays without placing a burden on the processor for these functions.

COUNTER

The 8-bit binary counter is presettable and readable with two MOV instructions which transfer the contents of the accumulator to the counter and vice versa. The counter content may be affected by RESET and should be initialized by software. The counter is stopped by a RESET or STOP TCNT instruction and is started by a START instruction or as an event counter by START CNT instruction. Once started the counter will increment to this maximum count (FF) and

overflow to zero continuing its count until stopped by a STOP TCNT instruction. The increment from maximum count to zero results in the setting of an overflow flag flip-flop and in the generation of an interrupt request. The state of the overflow flag is testable with the conditions jump instruction JTF. The flag is reset by executing a JTF or by RESET. The timer interrupt may be enabled or disabled independently of external interrupt by the ENT CNT1 and DISCNT1 instruction. If enabled, the counter overflow will cause a subroutine call to location 7 where the timer or counter service routine may be stored. If the timer and external interrupt occur simultaneously, the external source will be recognized and the call will be to location 3. Since the timer interrupt is lathed it will remain pending until the external device is serviced and immediately be recognized upon return from the services routine. The pending timer interrupt is reset by the call to location 7 or may be removed by executing a DISCNT1 instruction.

EVENT COUNTER

The counter input is connected with the T1 I/P pin as START CNT instruction is executed and the counter is enabled. The 1 input is sampled at the beginning of state 3. Subsequent high to low transition on T1 will cause the

counter to increment T1 must be held for atleast one T cycle to ensure that it wont be missed. The maximum rate at which the counter may be incremented is one per three instruction cycles. There is no minimum frequency T1 input must remain high alteast 1/5 of a cycle after transition.

TIMER

Execution of a START instruction connects an internal clock to the counter input and enables the counter. The internal clock is derived by passing the basic machine cycle clock through a 32 pre-scaler. The pre-scalar is reset during START T instruction. The resulting clock increments the counter every 32 machine cycle. Various delay from 1 to 256 counter can be obtained by presetting the counter and detecting overflow. ALE divided by 3 or more can serve as an external clock. Very small detage of 'fine timing' of larger delays can be easily accom||lished by software delay loops.

5.4 PROGRAMMING THE 8748

The chip details and the functional block diagram of 8748 have been given in Appendix (b).

The following chapter gives a brief overview of the system controller (8748) architecture and its design aspects.

The MCS -48 instruction set is extensive for machine of its size and has been tailored to be straight forward and very efficient in its use of program memory. All instructions are either one of two bytes in length and over 80% are only one byte long. Also, all instructions execute in either one or two cycles and over 50% of all instructions execute in a single cycle. Double cycle instructions include all immediate instructions, and all I/O instructions.

The MCS-48 microcomputers have been designed to handle arithmetic operations efficiently in both binary and BCD as well as handle the single-bit operations required in control applications. Special instructions have also been included to simplify loop counters, table look-up routines, and N-way branch routines.

DATA TRANSFERS

The 8-bit accumulator is the central point for all data transfers within the 8048. Data can be transferred between the 8 registers of each working register bank and the accumulator directly, i.e., the source or destination register is specified by the instruction. The remaining locations of the internal RAM array are referred to as Data Memory and are addressed indirectly via an address stored in

either R0 and R1 are also used to indirectly address external data memory when it is present. Transfers to and from internal RAM require one cycle, while transfers to external RAM require two. Constants stored in program Memory can be loaded directly to the accumulator and to the 8 working registers. Data can also be transferred directly between the accumulator and the program status word (PSW). Writing to the PSW alters machine status accordingly and provides a means of restoring status after an interrupt or of altering the stack pointer if necessary.

ACCUMULATOR OPERATIONS

Immediate data, data memory or the working registers can be added with or without carry to the accumulator. These sources can also be ANDed, ORed, or Exclusive ORed to the accumulator. Data may be moved to or from the accumulator and working registers or data memory. The two values can also be exchanged in a single operation.

In addition, the lower 4 bits of the accumulator can be exchanged with the lower 4-bits of any of the internal RAM locations. This instruction, along with an instruction which swaps the upper and lower 4-bit halves of the accumulator, provides for easy handling of 4-bit quantities, including BCD

numbers. To facilitate BCD arithmetic, a Decimal Adjust instruction is included. This instruction is used to correct the result of the binary addition of two 2-digit BCD numbers. Performing a decimal adjust on the result in the accumulator produces the required BCD result.

Finally, the accumulator can be incremented, decremented, cleared, or complemented and can be rotated left or right I bit at a time with or without carry.

Although there is no subtract instruction in the 8048AH, this operation can be easily implemented with three single byte single-cycle instruction.

A value may be subtracted from the accumulator with the result in the accumulator by:

Complementing the accumulator
Adding the value to the accumulator
complementing the accumulator

REGISTER OPERATIONS

The working registers can be accessed via the accumulator as explained above, or can be loaded immediate with constants from program memory. In addition, they can be incremented or

decremented or used as loop counters using the decrement and jump, if not zero instruction, as explained under branch instructions.

All Data Memory including working registers can be accessed with indirect instructions via R0 and R1 and can be incremented.

FLAGS

There are four user-accessible flags in the 8048AH; carry, Auxiliary carry, F0, and F1. Carry indicates overflow of the accumulator, and Auxiliary carry is used to indicate overflow between BCD digits and is used during decimal adjust operation. Both carry and Auxiliary carry are accessible as part of the program status word and are stored on the stack during subroutines. F0 and F1 are undedicated general-purpose flags to be used as the programmer desires. Both flags can be cleared or complemented and tested by conditional jump instructions. F0 is also accessible via the program status word and is stored on the stack with the carry flags.

BRANCH INSTRUCTIONS

The unconditional jump instruction is two bytes and allows jumps anywhere in the first 2K words of program

memory. Jumps to the second 2K of memory (4K words are directly addressable) are made first by executing a select memory bank instruction, then executing the jump instruction. The 2K boundary can only be crossed via a jump or subroutine call instruction, i.e., the bank switch does not occur until a jump is executed. Once a memory bank has been selected all subsequent jumps will be to the selected bank until another select memory bank instruction is executed. A subroutine in the opposite bank can be accessed by a select memory bank instruction followed by a call instruction. Upon completion of the subroutine, execution will automatically return to the original bank; however, unless the original bank is reselected, the next jump instruction encountered will again transfer execution to the opposite bank.

Conditional jumps can test the following inputs and machine status.

To Input pin

TI Input pin

INI Input pin

Accumulator Zero

Any bit of Accumulator

Carry Flag

F0 Flag

F1 Flag.

Conditional jumps allow a branch to any address within the current page (256 words) of execution. The conditions tested are the instantaneous values at the time the conditional jump is executed. For instance, the jump on accumulator zero instruction tests the accumulator itself not an intermediate zero flag.

The decrement register and jump if not zero instruction combines a decrement and a branch instructions to create an instruction very useful implementing a loop counter. This instruction can designer any one of the working registers as a counter and can elect a branch to any address within the current page of execution.

A single-byte indirect jump instruction allows the program to be vectored to anyone of several different locations based on the contents of the accumulator. The contents of the accumulate points to a location to program memory which contains the jump address. The 8-bit jump address refers to the current page of execution. This instruction could be used. For instance, to vector to any one of several routines based on an ASCII character which has been loaded in the accumulator to this way ASCII key key inputs can be used to initiate various routine.

SUBROUTINES

Subroutines are entered by executing a call instruction. Calls can be made like unconditional jumps to any address in a 2K word bank, and jumps across the 2K boundary are executed in the same manner. Two separate return instruction determine whether or not status (upper 4-bits of PSW) is restored upon return from the subroutine.

The return and restore status instruction also signals the end of an interrupt service routine if one has been in progress.

TIMER INSTRUCTIONS

The 8-bit on board timer counter can be loaded or read via the accumulator while the counter is stopped or while counting. The counter can be started as a timer with an external clock applied to the TI input pin. The instruction executed determines which clock source is used. A single instruction stops the counter whether it is operating with an internal or an external clock source. In addition, two instructions allow the timer interrupt to be enabled or disabled.

CONTROL INSTRUCTIONS

Two instructions allow the external interrupt source to be enabled or disabled. Interrupts are initially disabled and are automatically disabled while an interrupt service routine is in progress and re-enabled afterward.

There are four memory bank select instructions, two to designate the active working register bank and two to control program memory banks. The working register bank switch instructions allow the programmer to immediately substitute second 8-register working register bank for the one is use. This effectively provides 16 working registers or it can be used as a means of quickly saving the contents of the registers in response to an interrupt. The user has the option to switch or not to switch banks on interrupt. However, if the banks are switched, the original bank will be automatically restored upon execution of a return and restore status instruction at the end of the interrupt service routine.

A special instruction enables an internal clock, which is the XTAL frequent divided by three to be output on pin TO. This clock can be used as a general purpose clock in the user's system. The instruction should be used only to

initialize the system since the clock output can be disabled only by application of system reset.

INPUT / OUTPUT INSTRUCTIONS

Ports 1 and 2 are 8-bit static 1/0 ports which can be loaded to and from the accumulator. Outputs are statiscally latched but inputs are not latched and must be read while inputs are present. In addition, immediate data from program memory can be ANDed or ORed directly to port 1 and port 2 with the result remaining on the port. This allows "masks" stored in program memory to selectively set or reset individual bits of the I/O ports. Ports 1 and 2 configured to allow input on a given pin by first writing a "I" out to the pin.

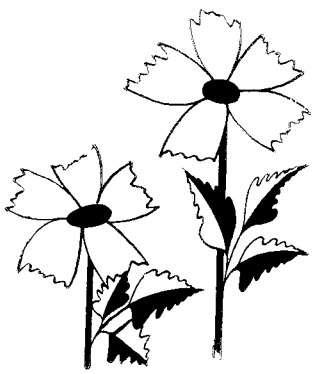
An 8-bit port called BUS can also be accessed via the accumulator and can have statiscally latched outputs as well. It too can have immediate data ANDed ORed directly to its outputs, however, unlike ports 1 and 2, all eight lines of BUS must be treated as either input or output at any one time. In additions to being a static port, BUS can be used as a true synchronous bi-directional port using the Move External instructions used to access external data memory. When these instructions are executed, a corresponding READ or

WRITE pulse is generated and data is valid only at that time. When data is not being transferred. BUS is in a high impedance state. Note that the OUTL, ANL, and the BRL instructions for the BUS are for use with internal memory only.

The basic three on-board I/O ports can be expanded via a 4-bit expander bus using half of port 2.I/O expander devices on this bus consist of 4-bit ports which are addressed as ports 4 through 7. These ports have their own AND and OR instructions like the on-board ports as well as move instructions to transfer data in or out. The expander AND and OR instructions, however, combine the contents of accumulator with the selected port rather than immediate data as is done with the on-board ports. I/O devices can also be added externally using the BUS port as the expansion bus. In this case the I/O port become "memory mapped", i.e., they are addressed in the same way as external data memory and exist in the external data memory address space addressed by pointer register R0 or R1.

The instruction set has been given in the Appendix (d).

The chip details and the functional block diagram of 8748 have been given in Appendix (b).



Software Implementation

CHAPTER VI

SOFTWARE

The software using microcontroller for timer/counter/RPM indicator applications is enclosed. The software is written in assembly language.

MODE 0 :

In this mode the LKT is pointed to the first data and output on ports. This pointer is incremented after a delay. After 8 outputs it again waits till the key is pressed and again continues.

MODE 1 :

In this mode R7 is treated as counter. The delay routines are switched over to RB1 bank. The Continue key is not pressed. The sequence is auto-repeated as MODE 0.

MODE 2 :

This mode is similar to other modes in delay routines. Here, the output energized is first taken into carry and the carry bit is set and again puts back to the accumulator and then put on ports. Thus the previous output is not cleared.

MODE 3 :

Here the look-up table has two data 00H and FFH. Delay is created. After a delay, the pointer will jump to the next data and the look-up table and output on the port.

MODE 4 : (Manual Mode)

In this mode the keys are sensed, after every output. If the key is pressed only the look-up table pointer is incremented. Otherwise it will be indefinitely waiting for the key. If stop key is pressed, it comes out of the loop.

MODE 5,6 :

In this counter mode the pulses are counted using the internal counter and then updated on the display depending on up/down. Up increments the display, down decrements the display.

MODE 7,8 :

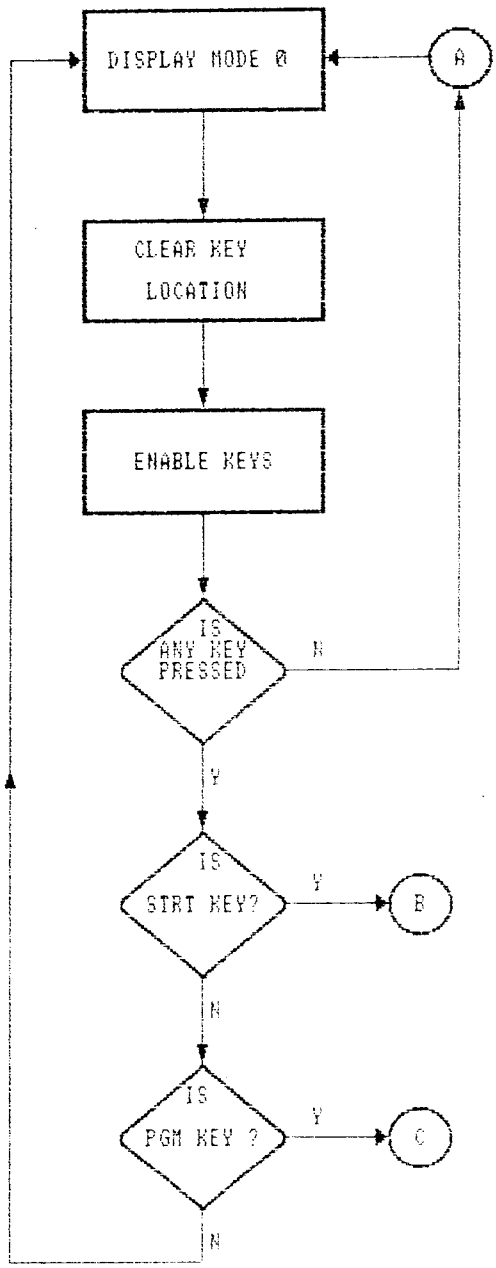
In this divider mode the pulses are counted as per the counter mode itself. But now the display gets incremented/decremented by comparing with the divisor.

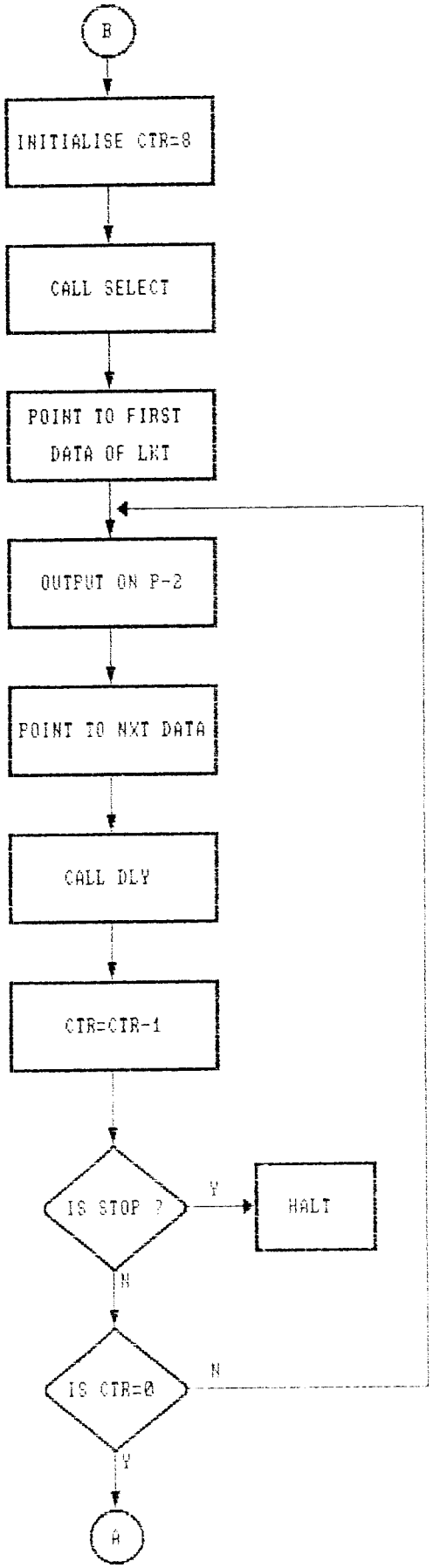
MODE 9 :

Here a time window opened for 1 sec. using the internal timer. During this time the pulses are counted and then multiplied by 60 and displayed. Thus this gives RPM.

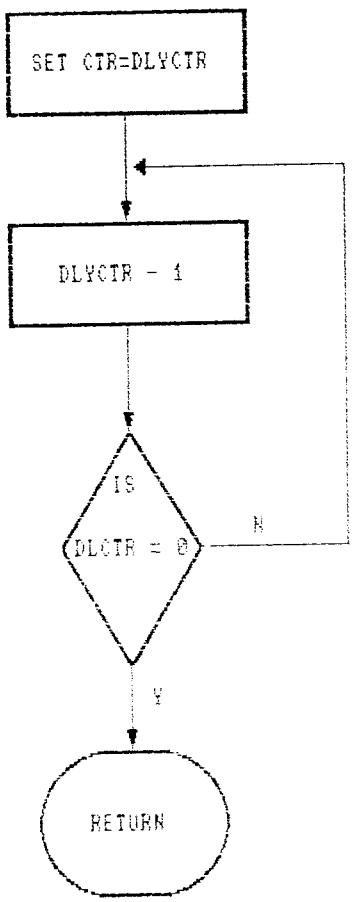
FLOW CHART

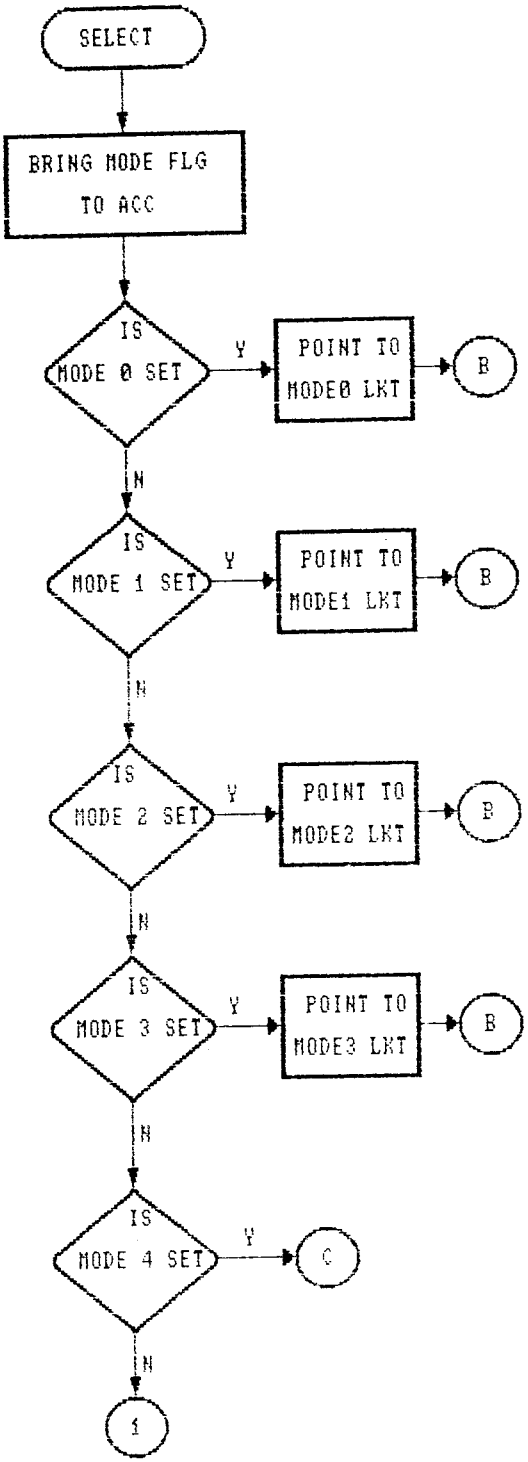

START
INITIALISE ALL PORT AS O/P PORTS
INTIALISE THE MODE TO MODE 0
CLEAR ALL TIMERS AND COUNTERS
CLEAR ALL DATA RAM LOCATIONS
CLEAR ALL DISPLAY SEGMENTS
A

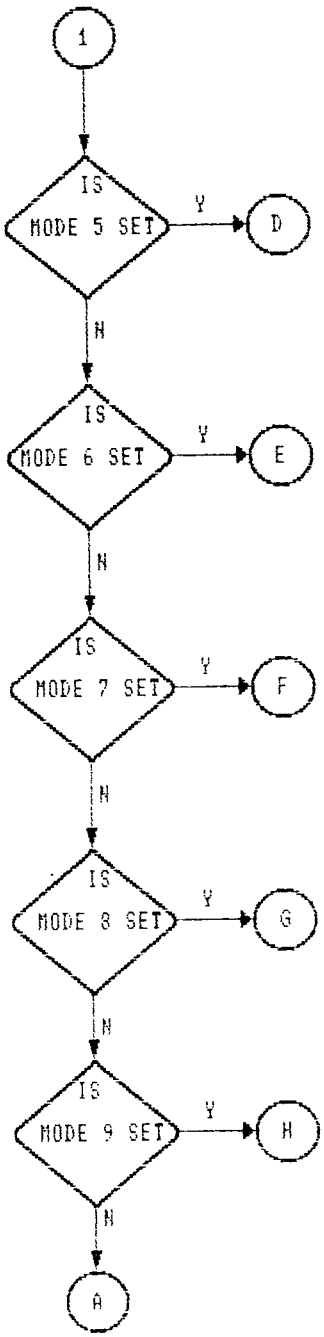

DISPLAY MODE 0
A
CLEAR KEY LOCATION
ENABLE KEYS
IS ANY KEY PRESSED
N
Y
IS STRT KEY?
Y
B
N
IS PGM KEY ?
Y
C
N


B
INITIALISE CTR=8
CALL SELECT
POINT TO FIRST DATA OF LKT
OUTPUT ON P-2
POINT TO NXT DATA
CALL DLY
CTR=CTR-1
IS STOP ?
Y
HALT
N
IS CTR=0
N
Y
A

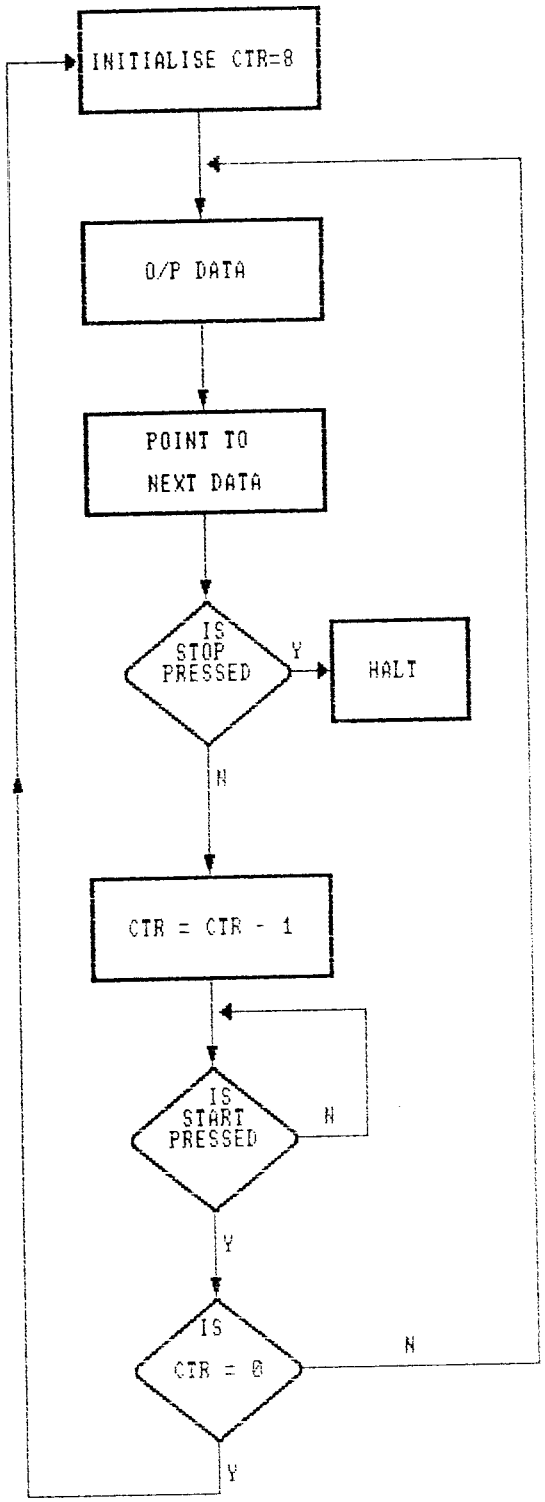
DELAY



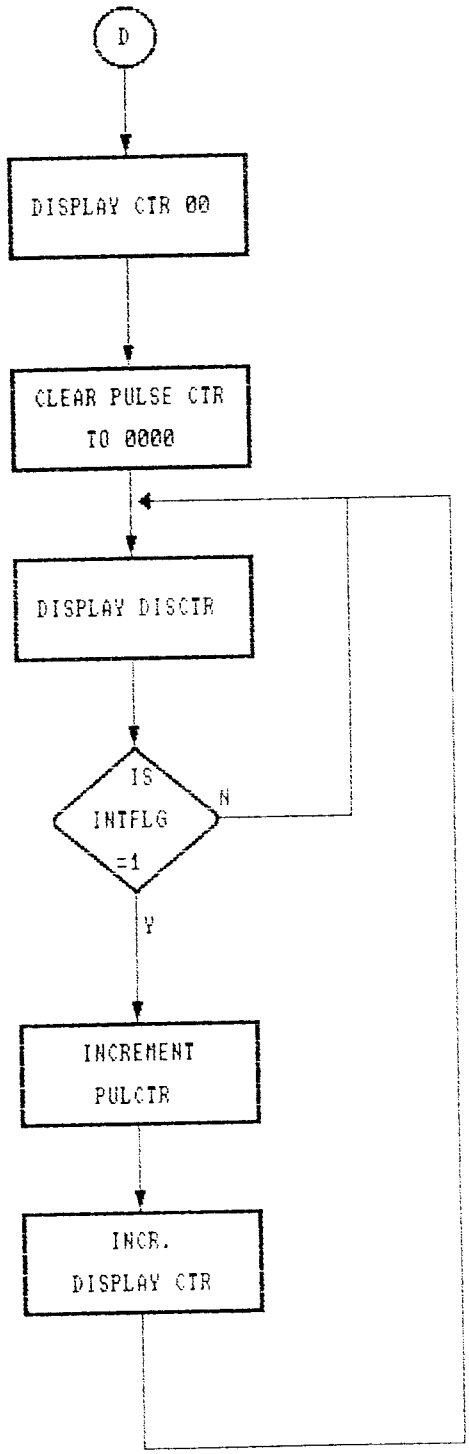

SELECT
BRING MODE FLG TO ACC
IS MODE 0 SET
Y
POINT TO MODE0 LKT
B
N
IS MODE 1 SET
Y
POINT TO MODE1 LKT
B
N
IS MODE 2 SET
Y
POINT TO MODE2 LKT
B
N
IS MODE 3 SET
Y
POINT TO MODE3 LKT
B
N
IS MODE 4 SET
Y
C
N
1


1
IS
MODE 5 SET
Y
D
N
IS
MODE 6 SET
Y
E
N
IS
MODE 7 SET
Y
F
N
IS
MODE 8 SET
Y
G
N
IS
MODE 9 SET
Y
H
N
A

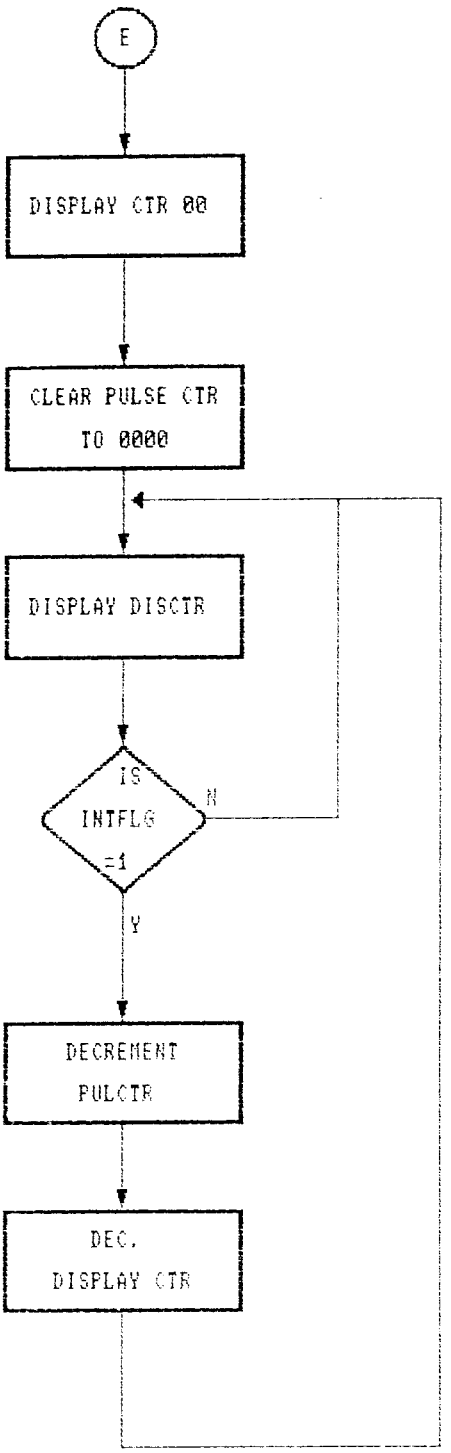
MANUAL MODE


INITIALISE CTR=8
O/P DATA
POINT TO
NEXT DATA
IS
STOP
PRESSED
Y
HALT
N
CTR = CTR - 1
IS
START
PRESSED
N
Y
IS
CTR = 0
N
Y

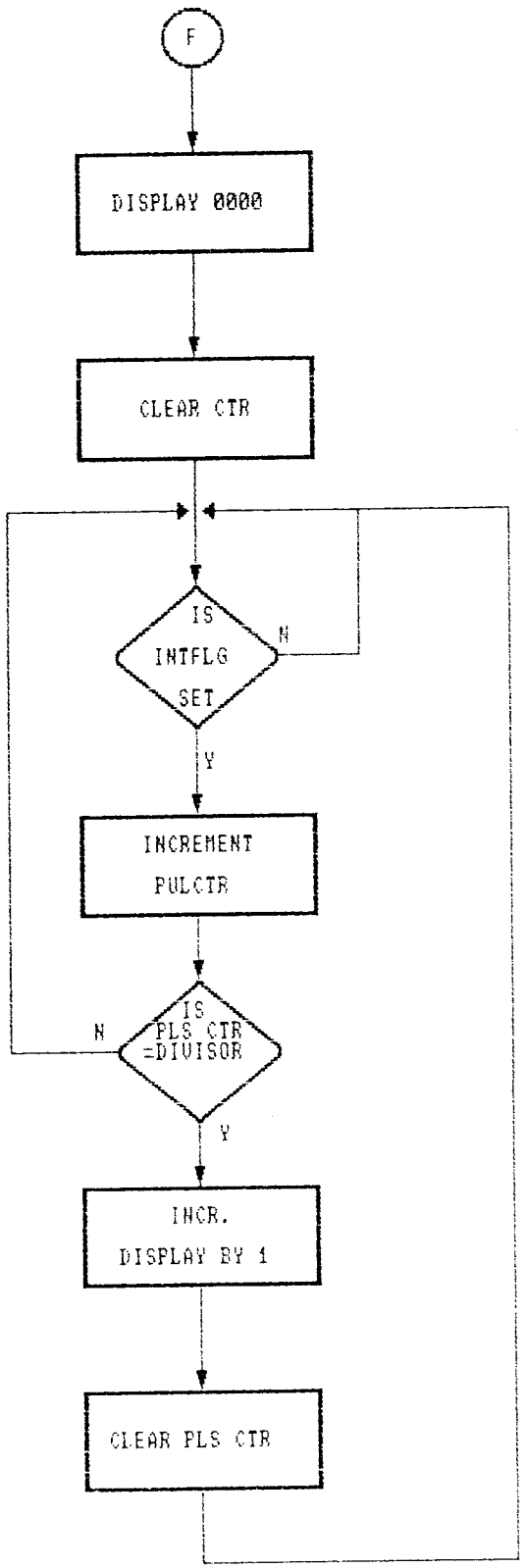
UP COUNTER


D
DISPLAY CTR 00
CLEAR PULSE CTR TO 0000
DISPLAY DISCTR
IS INTFLG =1
N
Y
INCREMENT PULCTR
INCR. DISPLAY CTR

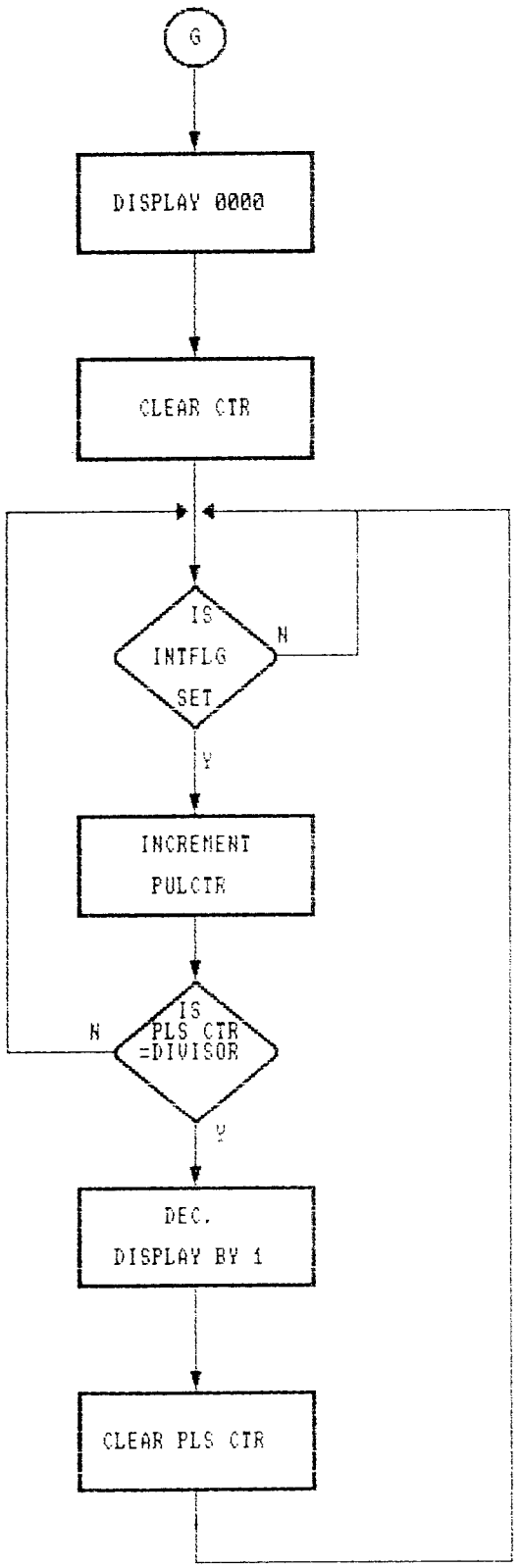
DOWN COUNTER



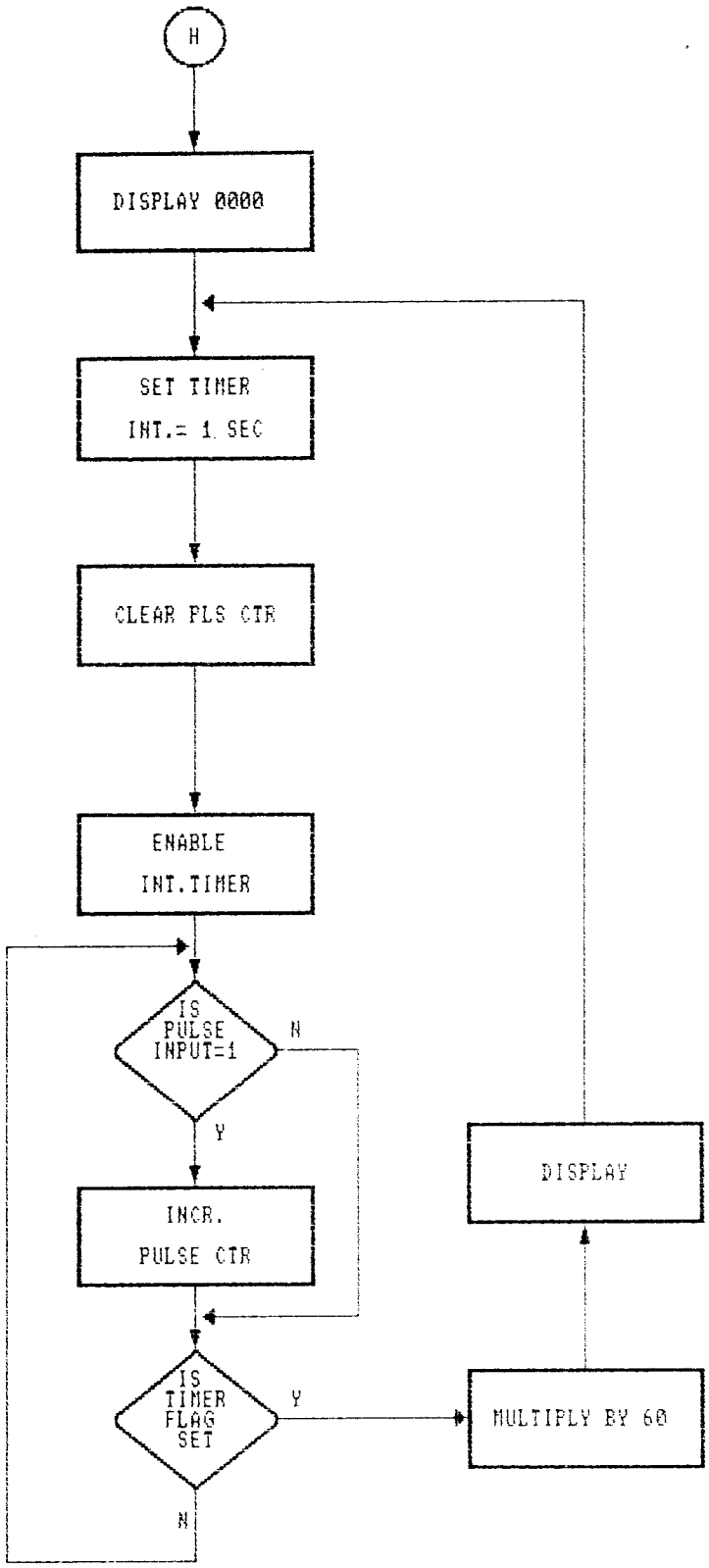
UP DIVIDER


F
DISPLAY 0000
CLEAR CTR
IS INTFLG SET
N
Y
INCREMENT PULCTR
IS PLS CTR =DIVISOR
N
Y
INCR. DISPLAY BY 1
CLEAR PLS CTR

DOWN DIVIDER



RPM


H
DISPLAY 0000
SET TIMER
INT.= 1 SEC
CLEAR PLS CTR
ENABLE
INT.TIMER
IS
PULSE
INPUT=1
N
Y
INCR.
PULSE CTR
IS
TIMER
FLAG
SET
Y
N
MULTIPLY BY 60
DISPLAY

```
;SOFTWARE FOR THE TIMER\COUNTER\RPM INDICATOR
;STUDENTS OF KCT
;ram equates follows
RAM STRT:    equ   20H          ;STARTING ADDRESS OGF THE INT RAM
CKDINP:      equ   RAMSTRT+1    ;stores the key code
RAWINP:      equ   CKDINP+1
MODEFLG:     equ   RAWINP+1     ;stores the mode of operation
SECCTR:      equ   MODEFLG+1
CNTFLG:      equ   SECCTR+2
PULCTR:      equ   CNTFLG+1
DIVISOR:     equ   PULCTR+2
OUTFLD:      equ   DIVISOR+1
DISBUF:      equ   OUTFLD+1
TEM_LOC:     equ   DISBUF+2
DLYCTR:      equ   TEM_LOC+2

;data ram equates follows
OUTCTR:      equ   08H
KEYENB:      equ   10111111B
KEYDIS:      equ   01000000B
DISCTR:      EQU   04H
DECPT:       EQU   01111111B
DIGTBL:      EQU   10H
ALPTBL:      EQU   1AH
SETVAL:      EQU   01H
M_TBL:       EQU   30H
OFST:        EQU   00H
UNUSED:      EQU   00011111B
;]display codes
;]                          hgfedcba          OLD BOARD NO

ZERO:        EQU   44H
ONE:         EQU   5FH
TWO:         EQU   2CH
THREE:       EQU   1CH
FOUR:        EQU   17H
FIVE:        EQU   94H
SIX:         EQU   84H
SEVEN:       EQU   5EH
EIGHT:       EQU   04H
NINE:        EQU   14H
BLANK:       EQU   FFH


;]transistor sel codes

DIGIT_1:     EQU   11011111B


DIGIT_2:     EQU   11101111B
DIGIT_3:     EQU   11110111B
DIGIT_4:     EQU   11111011B
DIGIT_5:     EQU   11111101B
DIGIT_6:     EQU   11111110B
```

```
LETR_f:     EQU   45H
LETR_n:     EQU   8FH
LETR_D:     EQU   FFH

            org       00h
            JMP       open

            org       03h
            JMP       exint

            org       07h
            JMP       TIMISR

;THIS IS THE START OF THE PROGRAM

open:       DIS       I
            DIS       TCNTI
            ORL       BUS,#FFH          ;DISABLE ALL INTERRUPTS
            MOV       R0,#MODEFLG
            MOV       @R0,#01H          ;DEFAULT ID MODE 0
            MOV       R0,#DLYCTR        ;CLEAR DLYCTR
            MOV       @R0,#00H
            INC       R0
            MOV       @R0,#00H
            JMP       DEFAULT

;now it begins !!!!

DEFAULT:    CALL      DLYDEF
            MOV       R0,#PULCTR
            MOV       @R0,#00H
            INC       R0
            MOV       @R0,#00H

MODECH:     MOV       R0,#MODEFLG       ;SET HE FLAG OF THE MODE
            MOV       @R0,#01H
            CLR       C
MODECH1:    CALL      MODETER
            CALL      UPADD
            CALL      KEYSCAN           ;CHECK THE KEYS FOR CLOSE
            MOV       R0,#CKDINP
            MOV       A,@R0
            JB0       BEGIN             ;DLY DISP
            JB1       MODEINC           ;MODE=MODE+1
            JB2       RUN               ;WAIT FOR START
            JMP       MODECH1
```

```
BEGIN:  MOV     R0,#DLYCTR
        CALL    DECODE
        CALL    UPADD
        CALL    KEYSCAN

        MOV     R0,#CKDINP
        MOV     A,@R0
        JB1     MODECH
        JB2     DLYINC
        JB3     DLYDEC
        JB0     MODECH1
        JMP     BEGIN

MODE0:  ;this mode gives o\ps sequentially and waits
        ;for the continue key to be pressed
        ;when con key is pressed the sequence resumes
        ;reset will reset the system
        ;stop will temporarily stop and start will
        ;commence from the stopped output
        ;THE PREVIOUS O\P ARE CLEARED DURING SUCCEDING
        CALL    KEYSCAN
        MOV     R0,#CKDINP
        MOV     A,@R0
        JB2     STM0
        JB1     MODECH1
        CALL    MODETFR
        CALL    UPADD
        JMP     MODE0
STM0:   ANL     P2,#FFH
        MOV     R5,#OUTCTR

        MOV     R0,#OUTFLD
        MOV     @R0,#FEH

LOOP0:  MOV     R0,#OUTFLD
        MOV     A,@R0
        OUTL    P2,A
        RL      A
        MOV     @R0,A
        CALL    MODETFR
        CALL    UPADD
        CALL    DELAY
        CALL    KEYSCAN
        MOV     R0,#CKDINP
        MOV     A,@R0
        JB4     STOPP
        ORL     P2,#FFH
        DJNZ    R5,LOOP0
        MOV     R0,#MODEFLG
        MOV     @R0,#01H
        JMP     MODE0
```

```
STOPP:      JMP     STOP
MODE1:      ;this mode is same as the mode0
            ;but the sequence is continued automatically
            ;THE PREVIOUS O\P CLEARED DURING SUCCEEDING O\PS

STM1:       ANL     P2,#FFH
            MOV     R7,#OUTCTR
            MOV     R0,#OUTFLD
            MOV     @R0,#FEH

LOOP1:      MOV     R0,#OUTFLD
            MOV     A,@R0
            OUTL    P2,A
            RL      A
            MOV     @R0,A
            CALL    MODETFR             ;fn-1 data are taken
            CALL    UPADD
            CALL    DELAY
            CALL    KEYSCAN
            MOV     R0,#CKDINP
            MOV     A,@R0
            JB4     STOPP

            ORL     P2,#FFH
            DJNZ    R7,LOOP1            ;STM1

MODE2:      ;this mode energizes outputs sequencially
            ;an automatic cycle is followed
            ;THE PREVIOUS O\P WILL REMAIN SAME
            ;keys action are same as mode1

STM2:       ORL     BUS,#FFH
            ANL     P2,#FFH
            MOV     R5,#OUTCTR
            MOV     R0,#OUTFLD
            MOV     @R0,#FFH

LOOP2:      MOV     R0,#OUTFLD
            MOV     A,@R0
            OUTL    P2,A
            CLR     C
            RLC     A
            MOV     @R0,A
            CALL    UPADD
            CALL    DELAY
            DJNZ    R5,LOOP2
            CALL    KEYSCAN
            MOV     R0,#CKDINP
            MOV     A,@R0
            JB4     STOPG
            JMP     STM2
```

```
MODE3:      ;this mode outputs all at a time and
            ;repeats after delay

STM3:       MOV     R0,#OUTFLD
            MOV     @R0,#00H

LOOP3:      CALL    MODETER
            CALL    UPADD
            MOV     R0,#OUTFLD
            MOV     A,@R0
            OUTL    P2,A
            CPL     A
            MOV     @R0,A
            CALL    DELAY
            CALL    KEYSCAN
            MOV     A,@R0
            JB4     STOP6
            JMP     LOOP3

STOP6:      JMP     STOP

STM4_NP:    JMP     STM4_N

MODECH1P:   JMP     MODECH1

MODE4:      CALL    KEYSCAN
            MOV     R0,#CKDINP
            MOV     A,@R0
            JB2     STM4_NP
            JB1     MODECH1P
            CALL    MODETER
            CALL    UPADD
            JMP     MODE0

STM4:       ANL     P2,#FFH
            MOV     R0,#OUTFLD
            MOV     @R0,#FEH

STM4_N:     MOV     R0,#OUTFLD
            MOV     A,@R0
            OUTL    P2,A
            RL      A
            MOV     @R0,A
            CALL    MODETER                 ;from data are taken
            CALL    UPADD
            ORL     P2,#FFH
            MOV     R0,#MODEFLG
            MOV     @R0,#01H
            JMP     MODE4
            ORG     200H
```

```
KEYSCAN:    ORL     P1          ,#0FFH          ;put in i/p mode
            ANL     BUS         ,#KEYENB        ;enable of switches
            MOV     R0          ,#RAWINP
            IN      A,P1
            ANL     P1          ,#00011111B
            MOV     @R0         ,A
            ORL     BUS         ,#.not.KEYENB   ;disable reading
            ANL     P1          ,#00H           ;switch off segments
            RET

DELAY:      SEL     RB1
            MOV     R0,#DLYCTR+1
            MOV     A,@R0

            MOV     R3,A
            DEC     R0

DLOOP21:    MOV     A,@R0
            MOV     R7,A

DLOOP1:     MOV     R5,#A0H
DLOOP0:     SEL     RB0

TRY1:       CALL    UPADD                       ;CALL   KEYSCAN
            NOP                                 ;MOV    A,@R0
            NOP                                 ;JB4    STOP
            NOP
            NOP
            NOP
            SEL     RB1
            DJNZ    R5,DLOOP0
            DJNZ    R7,DLOOP1
            DJNZ    R3,DLOOP2
            SEL     RB0
            RET

DLOOP2:     MOV     R0,#DLYCTR
            JMP     DLOOP21

STOP:       CALL    UPADD
            ORL     P2,#FFH
            CALL    KEYSCAN
            MOV     R0,#CKDINP
            MOV     A,@R0
            JB3     RUNPP
            JMP     STOP

RUNPP:      JMP     RUN
            ORG     400H
```

```
CONTIN1:    MOV     R0,#MODEFLG
            MOV     A,@R0
            JB0     LOOP0P
            JB1     LOOP1P
            JB2     LOOP2P
            JB3     LOOP3P
;           JB4     LOOP4P
;           JB5     LOOP5P
            JMP     $

LOOP0P:     JMP     LOOP0
LOOP1P:     JMP     LOOP1
LOOP2P:     JMP     LOOP2
LOOP3P:     JMP     LOOP3
;LOOP4P:             JMP     LOOP4
;LOOP5P:             JMP     LOOP5

UPCNT:      MOV     R0,#PULCTR
            MOV     A,@R0
            ADD     A,#01H
            DA      A
            MOV     @R0,A
            JNC     QUIT
            INC     R0
            MOV     A,@R0
            ADD     A,#01H
            DA      A
            MOV     @R0,A
            JMP     QUIT

DNCNT       MOV     R0,#PULCTR
            MOV     A,@R0
            ADD     A,#99H
            DA      A
            MOV     @R0,A
            JC      QUIT
            INC     R0
            MOV     A,@R0
            ADD     A,#99H
            DA      A
            MOV     @R0,A
            JMP     QUIT


;LOOK-UP TABLE STARTS
org 300h
DB          ZERO,ONE,TWO,THREE,FOUR,FIVE,SIX,SEVEN,EIGHT,NINE

ORG         30FH
DB          BLANK
org         310h
```

```
DB              DIGIT_1,DIGIT_2,DIGIT_3,DIGIT_4,DIGIT_5,DIGIT_6
org 31ah
DB              LEETR_D,LETR_n,LETR_f

or 320
Mode_1 LKT      FE,FD,FB,F7,EF,DF,BF,7F
Mode_2 LKT      FE,FD,FB,F7,EF,DF,BF,7F
Mode_3          FE,FC,F8,F0,E0,C0,80,00
Mode_4          FF,00

RPM Mode_9      MOV     R0,# TIMVAL     ; 1 sec
                MOV     A,R0
                MOV     T,A
                STRT    T
                ENT     I

DISP0           JNTO    DISP
                MOV     R0,#PLSCTR
                INC     R0
                JTF     UPDATE

DISP            CALL    UPADD
                JMP     DISP0

UPADD           CALL    TRSFR
                JMP     DISP
```

Testing

CHAPTER VII

TESTING

7.1 TESTING OF THE MAIN BOARD

1. First the crystal oscillator is checked.
2. ALE is checked.
3. If both are alright the processor is O.K.
4. The reset for normal operation should be high. This is checked. If the reset is now the processor.
5. Next the pulses at DB0-DB3 is checked.
6. Now if it is O.K, the software execution for display is O.K.

7.2 TESTING DISPLAY SECTION

1. Remove the processor give a permanent low at DB0 the first digit is enabled.
2. Next remove the latch, give a permanent '0' at output and observe at the display. The corresponding segment has to glow.

The above procedure is checked for DB1, DB2, DB3. Checks spikes of key and enable at DB4. Checks output and outputs leads by giving external signal to O/P sections. Test for counter operation RPM indicator and timer operation.

7.3 COUNTER MODE

In this mode the pulses are given to the T1 pin of the controller. The T1 pin whenever taken as input acts as on event counter whenever T1 is given a pulse. The internal timer register gets incremented by one. Now the counter register is read and corresponding action takes place. Whether it is down count or up count.

Whenever T1 register gets overflow on interrupt is produced. This interrupt brings the counter to stop. By giving a value in the display 200, the unit was tested. The value gets decreased and finally reaches zero and it comes to an halt.

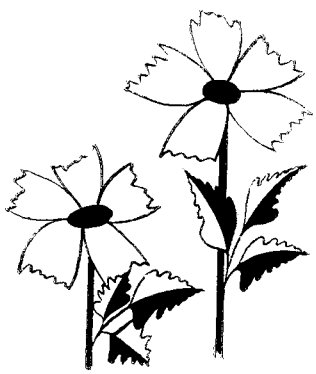
7.4 RPM MODE

To measure RPM the sensor which may be any photo effective or magnetic is used. Whenever the rotation starts the sensor gives out pulses. If these pulses are counted in a specified time periods, RPS is obtained.

RPS * 60 = RPM

7.5 TIMER MODE

When the start key is pressed, the timer will be loaded for a prescribed delay. After the delay is over an interrupt signal is obtained indicating the end of the set timing. Then the next value of time can be set.



Conclusion and Improvements

CHAPTER VIII

CONCLUSION

A micro controller based timer/counter/rpm indicator has been designed fabricated and tested. The instrument is found working satisfactorily.

This unit finds its applications in communication, industrial automation, process control, level finding etc., since Electronic counters are capable of making many measurement involving frequency, time, phase angle, and totalizing events. This unit can be used for such measurement.

Following improvements can be made in the unit.

1. The input of the output LED's for indication can be drives through a relay for the higher load.
2. This unit can be used for higher capacity by using suitable relays.
3. The 3-phase invertor driving circuit can be triggered by six outputs of timer using proper software.
4. The same hardware can be used as a hour counter for a machine.

References

REFERENCES

1. JOHN. B. PEATMAN, " DESIGN WITH MICROCONTROLLERS", Mc. GRAW HILL BOOK COMPANY, 1989.

2. "8 BIT EMBEDDED COUNTER", INTEL 1986.

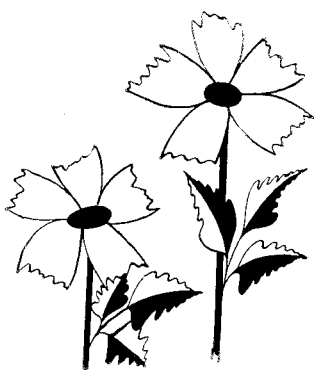
3. "8 BIT EMBEDDED COUNTER APPLICATION" INTEL.

4. "THE TTL DATA BOOK", TEXAS INSTRUMENTS, 1984.

5. "MICRO SYSTEM COMPONENTS HARD BOOK", INTEL 1986.

6. "MEMORY COMPONENTS HARDBOOK", INTEL SUPPLEMENT 1986.

7. SATNAM P. MATHUR, C. KULSHRESHTHA, R. CHADHA, "ELECTRONIC DEVICES APPLICATION AND INTEGRATED CIRCUITS", UMESH PUBLICATIONS 1989.



Appendix

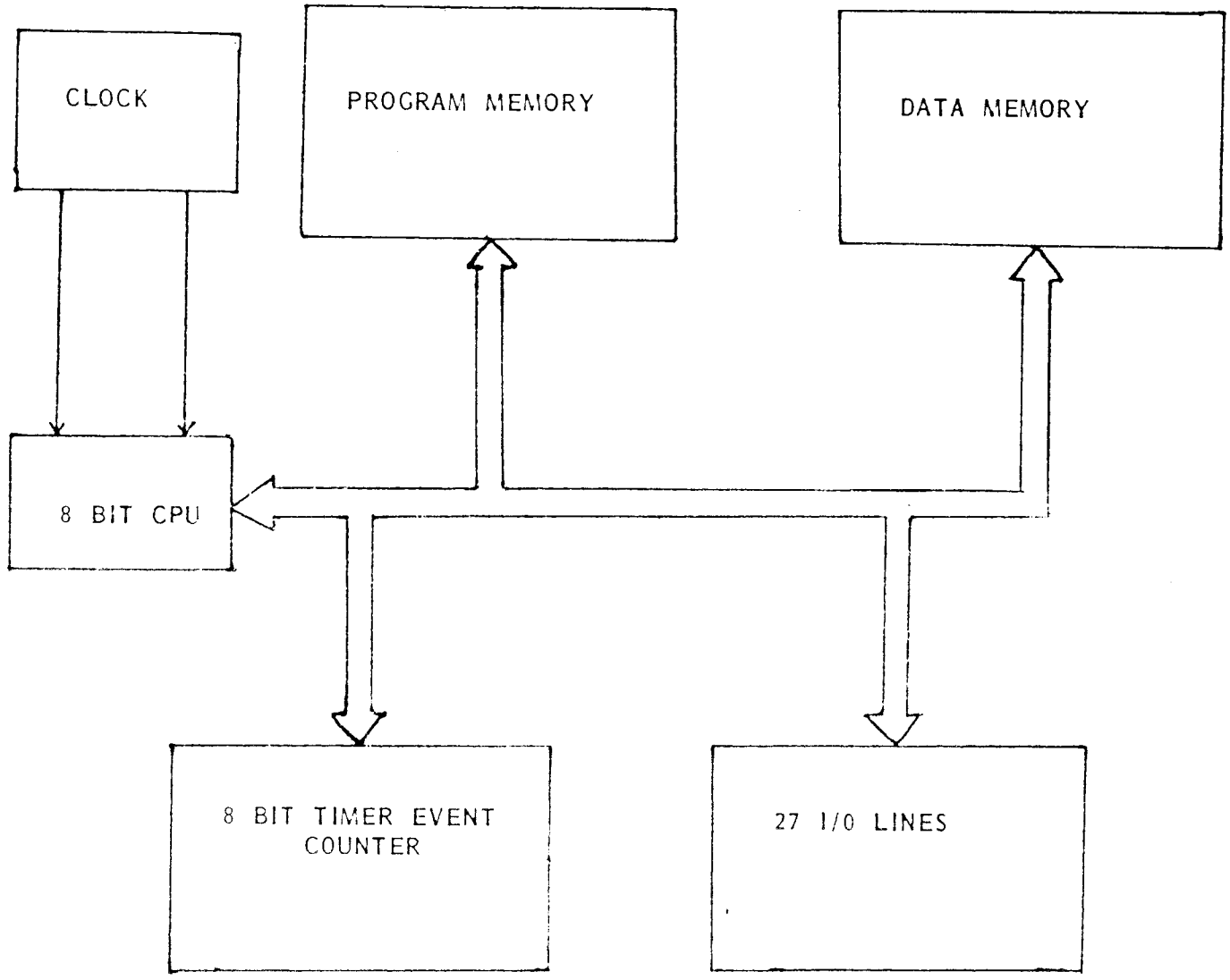


FIG: BLOCK DIAGRAM OF 8748 MICROCONTROLLER

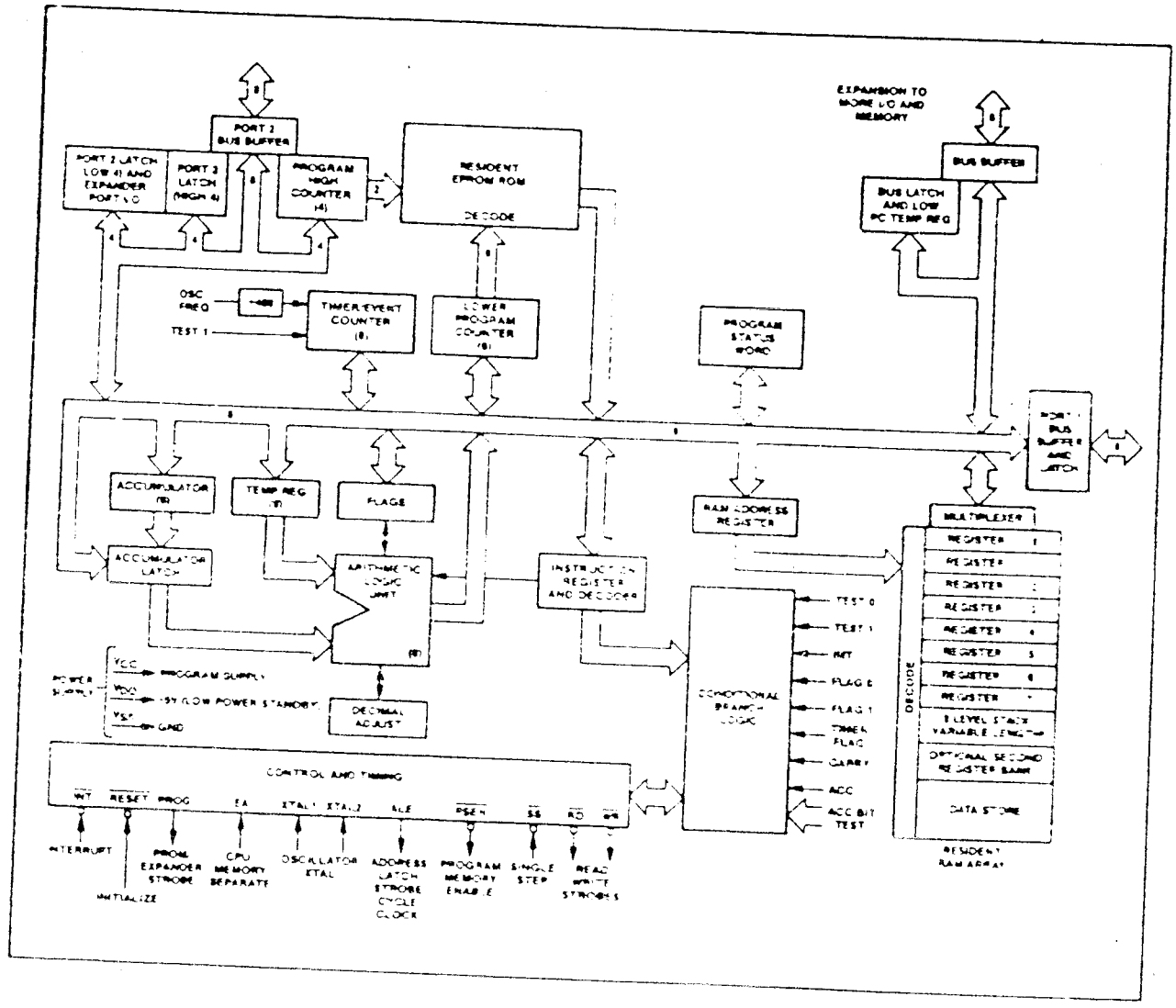


FIG: ARCHITECTURE OF 8748

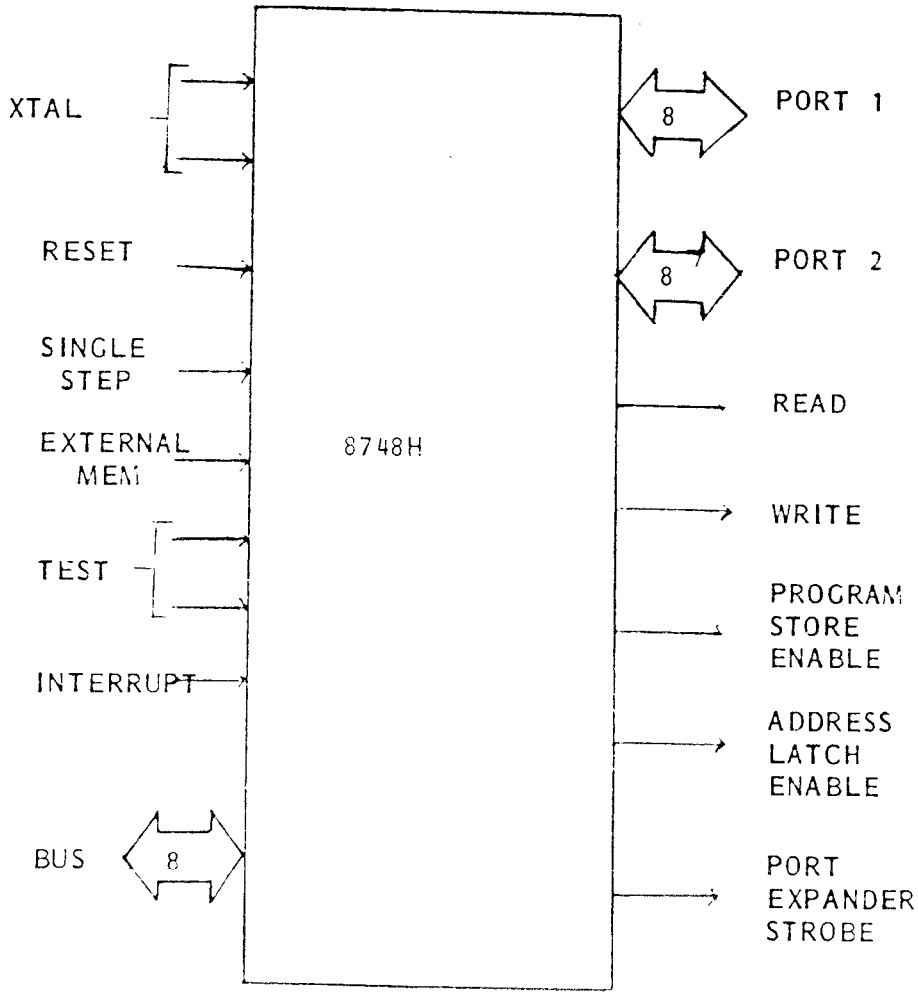


FIG:

LOGIC SYMBOL

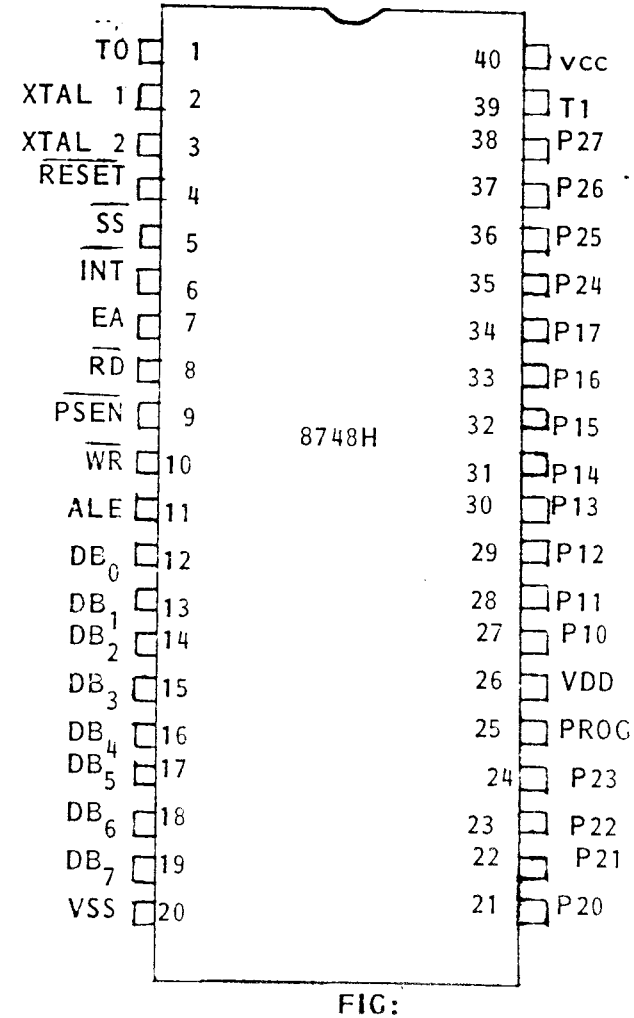


FIG:

PIN CONFIGURATION

Table 2. Instruction Set

Accumulator

Mnemonic	Description	Bytes	Cycles
ADD A, R	Add register to A	1	1
ADD A, @R	Add data memory to A	1	1
ADD A, # data	Add immediate to A	2	2
ADDC A, R	Add register with carry	1	1
ADDC A, @R	Add data memory with carry	1	1
ADDC A, # data	Add immediate with carry	2	2
ANL A, R	And register to A	1	1
ANL A, @R	And data memory to A	1	1
ANL A, # data	And immediate to A	2	2
ORL A, R	Or register to A	1	1
ORL A @R	Or data memory to A	1	1
ORL A, # data	Or immediate to A	2	2
XRL A, R	Exclusive or register to A	1	1
XRL A, @R	Exclusive or data memory to A	1	1
XRL, A, # data	Exclusive or immediate to A	2	2
INC A	Increment A	1	1
DEC A	Decrement A	1	1
CLR A	Clear A	1	1
CPL A	Complement A	1	1
DA A	Decimal adjust A	1	1
SWAP A	Swap nibbles of A	1	1
RL A	Rotate A left	1	1
RLC A	Rotate A left through carry	1	1
RR A	Rotate A right	1	1
RRC A	Rotate A right through carry	1	1

Input/Output

Mnemonic	Description	Bytes	Cycles
IN A, P	Input port to A	1	2
OUTL P, A	Output A to port	1	2
ANL P, # data	And immediate to port	2	2
ORL P, # data	Or immediate to port	2	2
INS A, BUS	Input BUS to A	1	2
OUTL BUS, A	Output A to BUS	1	2
ANL BUS, # data	And immediate to BUS	2	2
ORL BUS, # data	Or immediate to BUS	2	2
MOVD A, P	Input expander port to A	1	2
MOVD P, A	Output A to expander port	1	2
ANLD P, A	And A to expander port	1	2
ORLD P, A	Or A to expander port	1	2

Registers

Mnemonic	Description	Bytes	Cycles
INC R	Increment register	1	1
INC @R	Increment data memory	1	1
DEC R	Decrement register	1	1

Branch

Mnemonic	Description	Bytes	Cycles
JMP addr	Jump unconditional	2	2
JMPP @A	Jump indirect	1	2
DJNZ R, addr	Decrement register and skip	2	2
JC addr	Jump on carry = 1	2	2
JNC addr	Jump on carry = 0	2	2
JZ addr	Jump on A zero	2	2
JNZ addr	Jump on A not zero	2	2
JT0 addr	Jump on T0 = 1	2	2
JNT0 addr	Jump on T0 = 0	2	2
JT1 addr	Jump on T1 = 1	2	2
JNT1 addr	Jump on T1 = 0	2	2
JF0 addr	Jump on F0 = 1	2	2
JF1 addr	Jump on F1 = 1	2	2
JTF addr	Jump on timer flag	2	2
JNI addr	Jump on INT = 0	2	2
JBb addr	Jump on accumulator bit	2	2

Subroutine

Mnemonic	Description	Bytes	Cycles
CALL addr	Jump to subroutine	2	2
RET	Return	1	2
RETR	Return and restore status	1	2

Flags

Mnemonic	Description	Bytes	Cycles
CLR C	Clear carry	1	1
CPL C	Complement carry	1	1
CLR F0	Clear flag 0	1	1
CPL F0	Complement flag 0	1	1
CLR F1	Clear flag 1	1	1
CPL F1	Complement flag 1	1	1

Table 2. Instruction Set (Continued)

Data Moves

Mnemonic	Description	Bytes	Cycles
MOV A, R	Move register to A	1	1
MOV A, @R	Move data memory to A	1	1
MOV A, # data	Move immediate to A	2	2
MOV R, A	Move A to register	1	1
MOV @R, A	Move A to data memory	1	1
MOV R, # data	Move immediate to register	2	2
MOV @R, # data	Move immediate to data memory	2	2
MOV A, PSW	Move PSW to A	1	1
MOV PSW, A	Move A to PSW	1	1
XCH A, R	Exchange A and register	1	1
XCH A, @R	Exchange A and data memory	1	1
XCHD A, @R	Exchange nibble of A and register	1	1
MOVX A, @R	Move external data memory to A	1	2
MOVX @R, A	Move A to external data memory	1	2
MOVP A, @A	Move to A from current page	1	2
MOVP3 A, @A	Move to A from page 3	1	2

Timer/Counter

Mnemonic	Description	Bytes	Cycles
MOV A, T	Read timer/counter	1	1
MOV T, A	Load timer/counter	1	1
STRT T	Start timer	1	1
STRT CNT	Start counter	1	1
STOP TCNT	Stop timer/counter	1	1
EN TCNTI	Enable timer/counter interrupt	1	1
DIS TCNTI	Disable timer/counter interrupt	1	1

Control

Mnemonic	Description	Bytes	Cycles
EN I	Enable external interrupt	1	1
DIS I	Disable external interrupt	1	1
SEL RB0	Select register bank 0	1	1
SEL RB1	Select register bank 1	1	1
SEL MB0	Select memory bank 0	1	1
SEL MB1	Select memory bank 1	1	1
ENT0 CLK	Enable clock output on T0	1	1

Mnemonic	Description	Bytes	Cycles
NOP	No operation	1	1

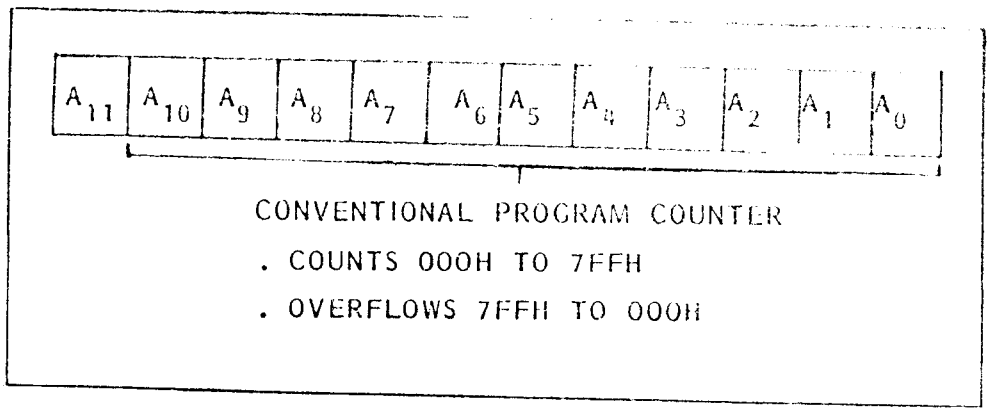


FIG: PROGRAM COUNTER

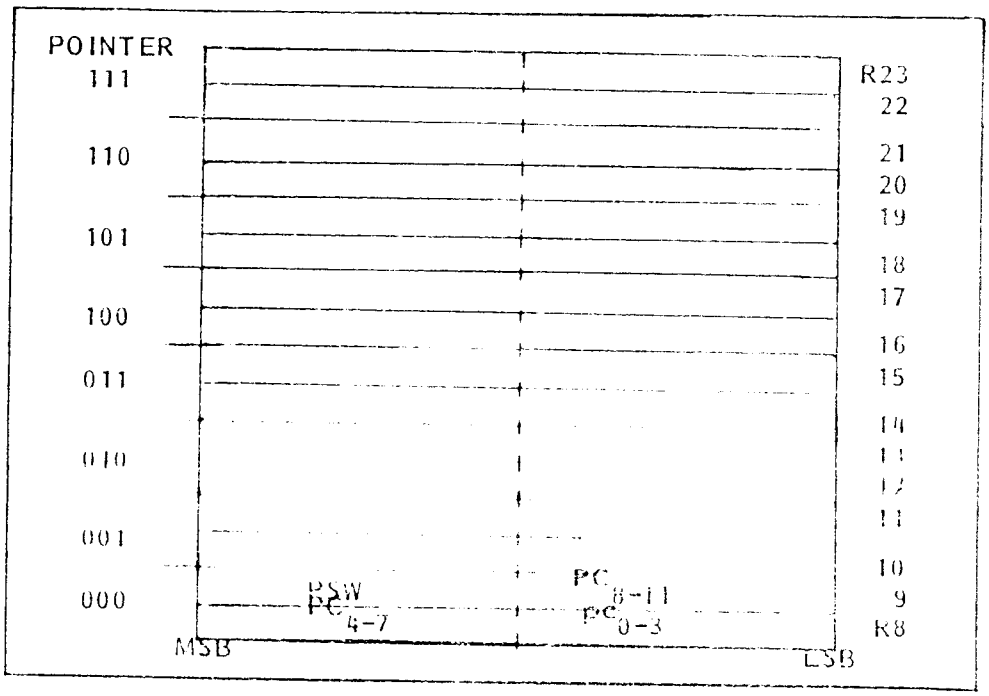


FIG: PROGRAM COUNTER STACK

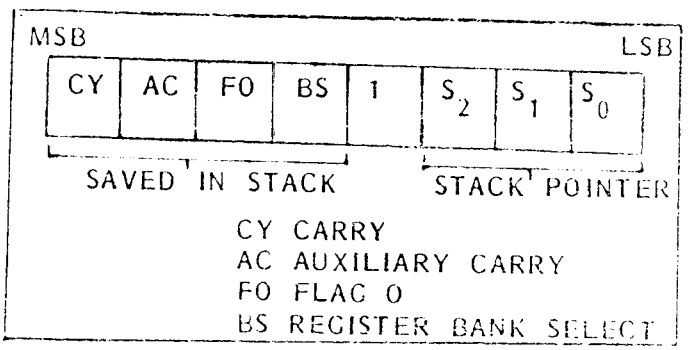


FIG: PROGRAM STATUS WORD(PSW)

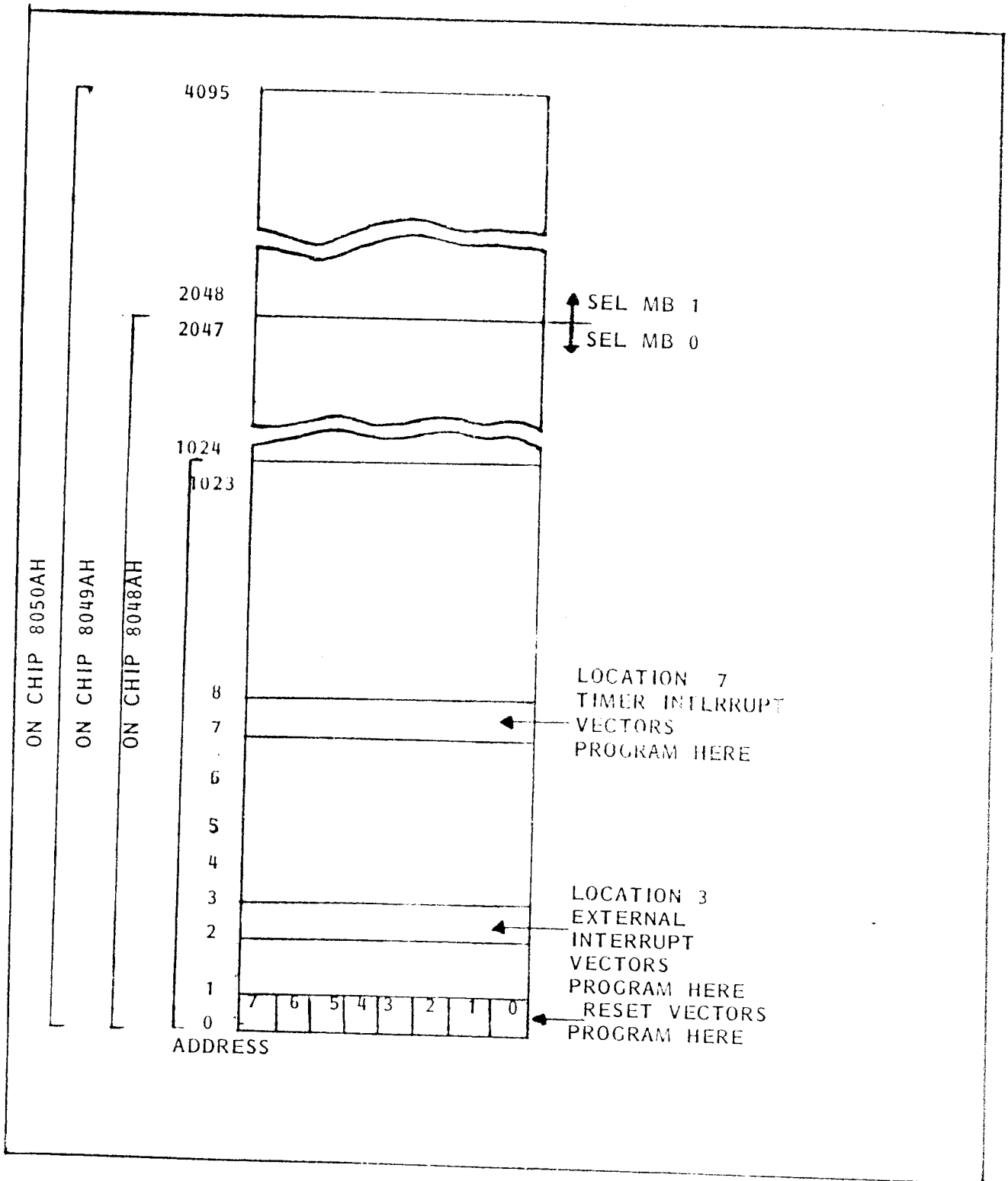


FIG: PROGRAM MEMORY MAP

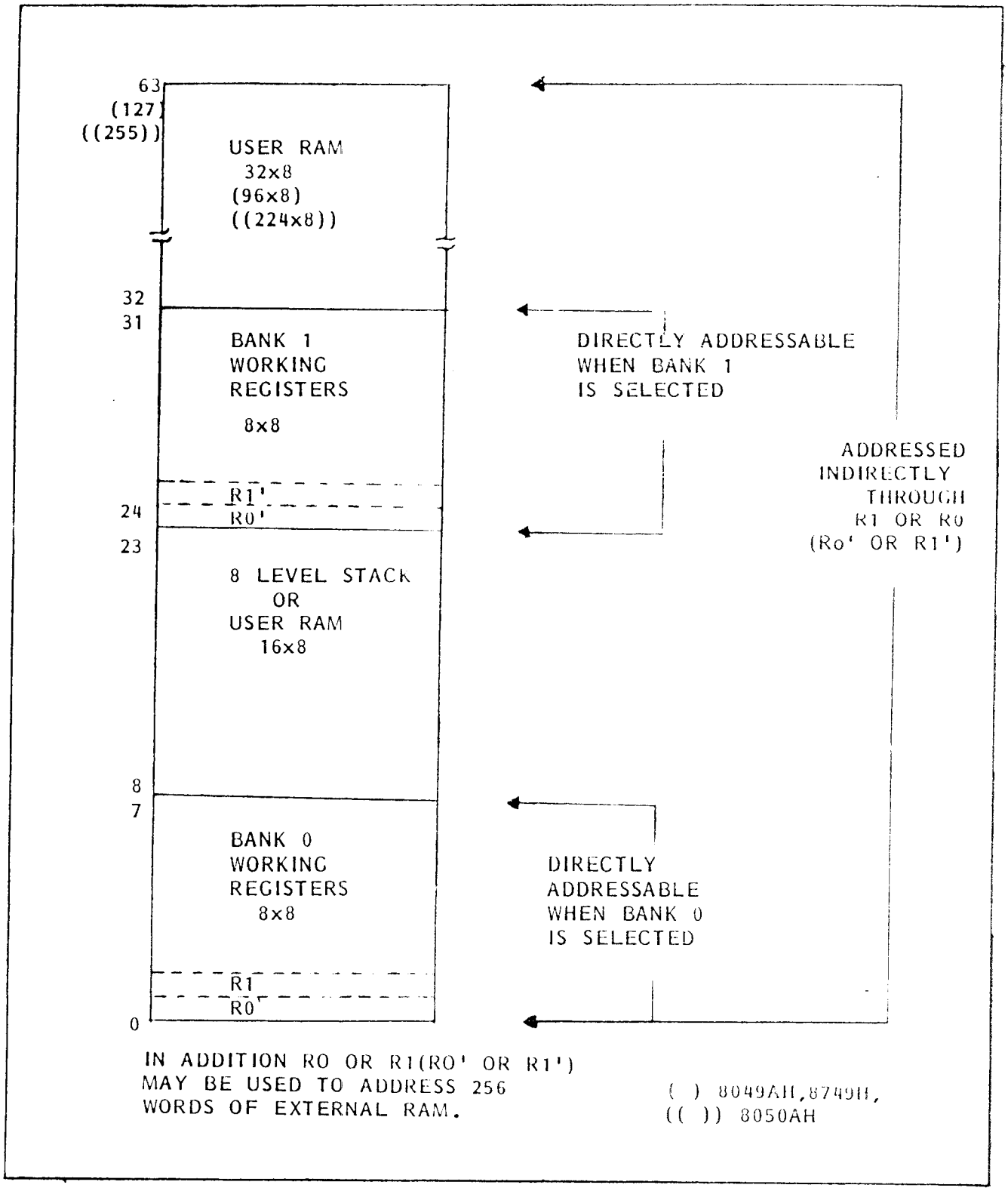
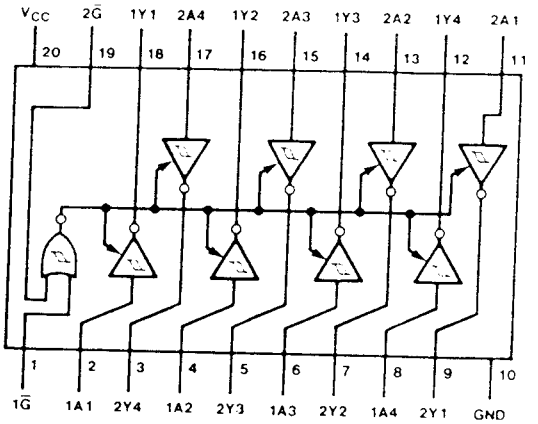
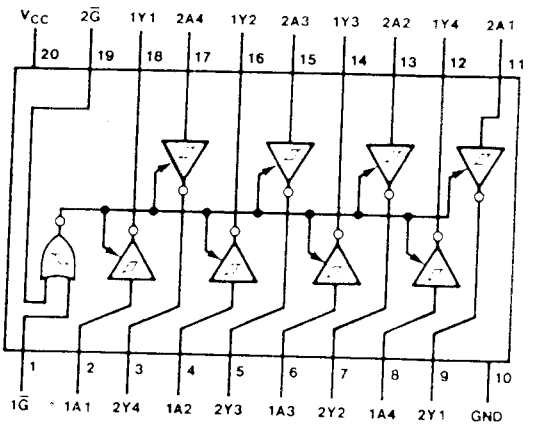


FIG: **DATA MEMORY MAP**

Connection Diagrams (Continued)



54S940 (J); 74S940 (N)



54S941 (J); 74S941 (N)

Octal TRI-STATE® Buffers/Line Drivers/Line Receivers

General Description

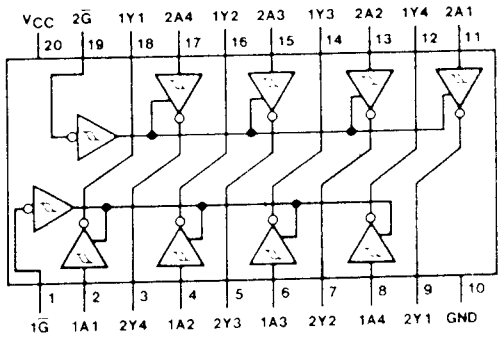
These buffers/line drivers are designed to improve both the performance and PC board density of TRI-STATE® buffers/drivers employed as memory-address drivers, clock drivers, and bus-oriented transmitters/receivers. Featuring 400 mV of hysteresis at each low current PNP data line input, they provide improved noise rejection and high fanout outputs, and can be used to drive terminated lines down to 133 Ω.

Features

- TRI-STATE outputs drive bus lines directly
- PNP inputs reduce DC loading on bus lines
- Hysteresis at inputs improves noise margins

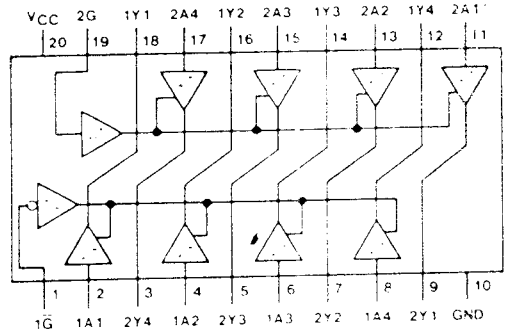
	Typical I_{OL} (Sink Current)	Typical I_{OH} (Source Current)	Typical Propagation Delay Times Inverting	Noninverting	Typical Enable/Disable Times	Typical Power Dissipation (Enabled) Inverting	Noninverting
74LS	12 mA	−1.2 mA	10.5 ns	[illegible]	[illegible]	130 mW	135 mW
74LS	24 mA	−15 mA	10.5 ns	[illegible]	[illegible]	130 mW	135 mW
74S	48 mA	−12 mA	4.5 ns	[illegible]	[illegible]	450 mW	538 mW
74S	64 mA	−15 mA	4.5 ns	[illegible]	[illegible]	450 mW	538 mW

Connection Diagrams



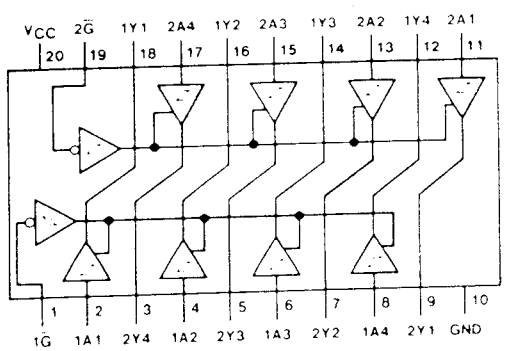
54LS240 (J)
54S240 (J)

74LS240 (N)
74S240 (N)



54LS241 (J)
54S241 (J)

74LS241 (N)
74S241 (N)



54LS244 (J)
54S244 (J)

74LS244 (N)
74S244 (N)

Switching Characteristics V_{CC} = 5 V, T_A = 25°C

Parameter		Conditions			DM54/74 LS240, LS241 LS244			DM54/74 S240, S241, S244 S940, S941			Unit
					Min	Typ (1)	Max	Min	Typ (1)	Max	
t_{pLH}	Propagation Delay Time Low to High Level Output	C_L = 45 pF	R_L = 667 Ω	LS240	3	9	14				ns
				LS241, 244	5	12	18				
			R_L = 90 Ω	S240, 940				2	4.5	7	
				S241, 244, 941				2	6	9	
t_{pHL}	Propagation Delay Time High to Low Level Output	C_L = 45 pF	R_L = 667 Ω	LS240	5	12	18				ns
				LS241, 244	7	12	18				
			R_L = 90 Ω	S240, 940				2	4.5	7	
				S241, 244, 941				2	6	9	
t_{pZL}	Output Enable Time to Low Level	C_L = 45 pF	R_L = 667 Ω	LS240	10	20	30				ns
				LS241, 244	10	20	30				
			R_L = 90 Ω	S240, 940				3	10	15	
				S241, 244, 941				3	10	15	
t_{pZH}	Output Enable Time to High Level	C_L = 45 pF	R_L = 667 Ω	LS240	5	15	23				ns
				LS241, 244	5	15	23				
			R_L = 90 Ω	S240, 940				2	6.5	10	
				S241, 244, 941				3	8	12	
t_{pLZ}	Output Disable Time from Low Level	C_L = 5 pF	R_L = 667 Ω	LS240	7	15	25				ns
				LS241, 244	8	15	25				
			R_L = 90 Ω	S240, 940				4	10	15	
				S241, 244, 941				2	10	15	
t_{pHZ}	Output Disable Time from High Level	C_L = 5 pF	R_L = 667 Ω	LS240	5	10	18				ns
				LS241, 244	5	10	18				
			R_L = 90 Ω	S240, 940				2	6	9	
				S241, 244, 941				2	6	9	
t_{pLH}	Propagation Delay Time Low to High Level Output	C_L = 150 pF	R_L = 667 Ω	LS240	5	11	18				ns
				LS241, 244	6	14	21				
			R_L = 90 Ω	S240, 940				3	7	10	
				S241, 244, 941				4	9	12	
t_{pHL}	Progapation Delay Time High to Low Level Output	C_L = 150 pF	R_L = 667 Ω	LS240	6	15	22				ns
				LS241, 244	6	15	22				
			R_L = 90 Ω	S240, 940				3	7	10	
				S241, 244, 941				4	9	12	
t_{pZL}	Output Enable Time to Low Level	C_L = 150 pF	R_L = 667 Ω	LS240	12	22	33				ns
				LS241, 244	12	22	33				
			R_L = 90 Ω	S240, 940				6	14	21	
				S241, 244, 941				6	14	21	
t_{pZH}	Output Enable Time to High Level	C_L = 150 pF	R_L = 667 Ω	LS240	6	18	26				ns
				LS241, 244	11	18	26				
			R_L = 90 Ω	S240, 940				4	9	12	
				S241, 244, 941					10	15	

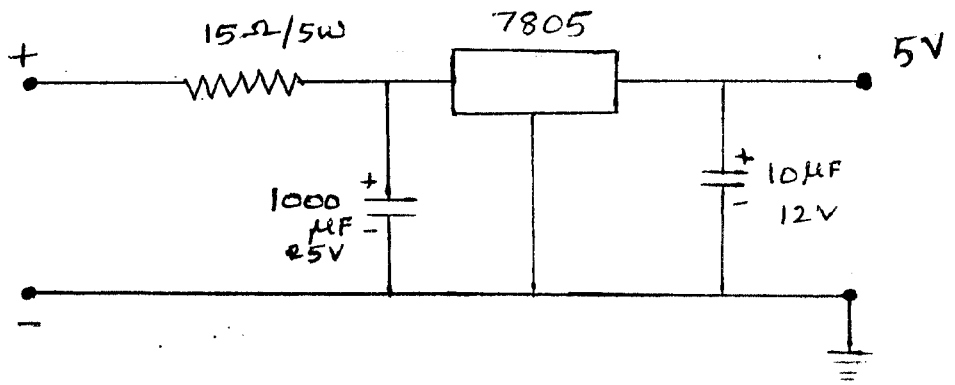
Electrical Characteristics over recommended operating free-air temperature range (unless otherwise noted)

Parameter		Conditions			DM54/74 LS240, LS241 LS244			DM54, 74 S240, S241, S244 S940, S941			Units
					Min	Typ (1)	Max	Min	Typ (1)	Max	
V_{IH}	High Level Input Voltage				2			2			V
V_{IL}	Low Level Input Voltage						0.8			0.8	V
V_I	Input Clamp Voltage	V_{CC} = Min, I_I = −18 mA					−1.5			−1.2	V
	Hysteresis ($V_{T+} - V_{T-}$)	V_{CC} = Min			0.2	0.4		0.2	0.4		V
I_{OH}	High Level Output Current			DM54			−12			−12	mA
				DM74			−15			−15	
V_{OH}	High Level Output Voltage	V_{CC} = 4.75 V, V_{IH} = 2 V, V_{IL} = 0.8 V, I_{OH} = −1 mA			2.7			2.7			V
		V_{CC} = Min, V_{IH} = 2 V, V_{IL} = 0.8 V, I_{OH} = −3 mA			2.4	3.4		2.4	3.4		
		V_{CC} = Min, V_{IH} = 2 V, V_{IL} = 0.5 V, I_{OH} = Max			2			2			
I_{OL}	Low Level Output Current			DM54			12			48	mA
				DM74			24			64	
V_{OL}	Low Level Output Voltage	V_{CC} = Min, V_{IL} = 0.8 V, V_{IH} = 2 V	I_{OL} = 12 mA				0.4				V
			I_{OL} = Max	DM54			0.4			0.55	
				DM74			0.5				
I_{OZH}	Off-State Output Current High Level Voltage Applied	V_{CC} = Max, V_{IL} = 0.8 V, V_{IH} = 2 V	V_O = 2.7 V				20				μA
			V_O = 2.4 V							50	
I_{OZL}	Off-State Output Current, Low Level Voltage Applied		V_O = 0.4 V				−20				
			V_O = 0.5 V							−50	
I_I	Input Current at Maximum Input Voltage	V_{CC} = Max	V_I = 7 V				0.1				mA
			V_I = 5.5 V							1	
I_{IH}	High Level Input Current	V_{CC} = Max, V_I = 2.7 V					20			50	μA
I_{IL}	Low Level Input Current		V_I = 0.4 V				−0.2				mA
		V_{CC} = Max	V_I = 0.5 V	Any A						−400	μA
				Any G						−2	mA
I_{OS}	Short Circuit Output Current	V_{CC} = Max (2)			−40		−225	−50		−225	mA
I_{CC}	Supply Current	V_{CC} = Max, Outputs Open	Outputs High	LS240, 241, 244		13	23				mA
				S240, S940	DM54				80	123	
					DM74				80	135	
				S241, 244, S941	DM54				95	147	
					DM74				95	160	
			Outputs Low	LS240		26	44				
				LS241, 244		27	46				
				S240, S940	DM54				100	145	
					DM74				100	150	
				S241, 244, S941	DM54				120	170	
					DM74				120	180	
			Outputs Disabled	LS240		29	50				
				LS241, 244		32	54				
				S240, S940	DM54				100	145	
					DM74				100	150	
				S241, 244, S941	DM54				120	170	
					DM74				120	180	

Note 1: All typical values are at V_{CC} = 5 V, T_A = 25°C.

Note 2: Not more than one output should be shorted at a time and duration should not exceed one second.

POWER SUPPLY CIRCUIT DIAGRAM.



PART LIST.

U_1 → MICRO CONTROLLER 8748

U_2 → LATCH 74HC373

U_3 → BUFFER 74LS244

$R_1 - R_8$ → RESISTERS 100-Ω

$R_{17} - R_{24}$ → " 100-Ω

R_{26-29} → " 10K

$R_{30} - R_{33}$ → " 2.7K

RESISTERS

R_9, R_{10}, R_{12}, R_{13}, R_{21}, are 1K

TRANSISTERS → 2N2907

Capacitors C_1 & C_2 → 20 PF

" C_3, C_4, C_5 → 0.1 MF

Ic → 7414

Ic → LM339

DIODE → IN4148