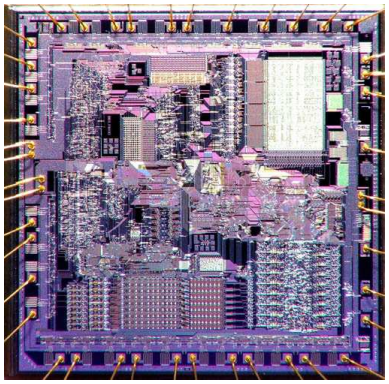
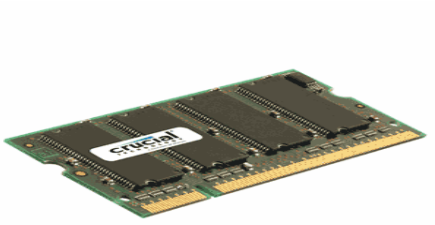
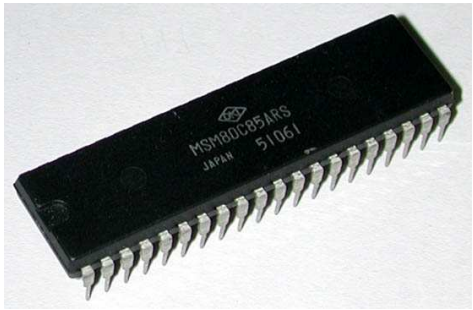


Designing a counter from 0 to 99 seconds





SUBMITTED BY

Name - Moumita Guha Roy
Roll No :- 3135-51-0017
Registration Number :- 019442
Session :- 2006-2007

Name - Dipendu Ghosh
Roll No :- 3135-61-0015
Registration Number :- 000439
Session :- 2006-2007

Acknowledgement

With great pleasure we would like to express our sincere gratitude to the respected teachers Prof. Diptiman Raichoudhuri, Department of Electronic Science, University of Calcutta and Prof. Seeraz Nawaz, Department of Computer Science, Vidyasagar College who have introduced us to these fascinating fields of study and guided us with constant encouragement for successful completion.

We also pleasured to acknowledge our indebt ness to Prof. Paramita Sikdar, Head of the Department of Computer Science, Vidyasagar College; Prof. Raj Kumar Panda, Department of Computer Science, Vidyasagar College; Mr. Sourov Saha, Department of Computer Science, Vidyasagar College; Mr. Bijon Kumar Pal, Department of Computer Science, Vidyasagar College and Mrs.Nilima Das for their help at various critical stages of the project.

We also wish to thank our friends for their cooperation and constant encouragement.

…….. …….

Moumita Guha Roy Dipendu Ghosh

Date:…….

Department of Computer Science Vidyasagar College

The students Dipendu Ghosh and Moumita Guha Roy have carried out the following project work under my supervision titled

"……
……."

During this project work they have learnt how to write Assembly Language Programming and how to implement them using the DYNA-85 kit. They have performed implementation of simple algorithms in Assembly Language Programming

I assure that this topic has not been taken up by any other student neither it has been given to any other student.

……..

Date:…… Mr. Seeraz Nawaz
(Guide)

……..

Date:…… (External)

INDEX

Introduction

The 8085 is a multipurpose, programmable, microprocessor chip that can perform different arithmetic and logical operations. Also it can be interfaced with different peripheral chips to do different kinds of jobs. Different peripheral chips can be interfaced like :-

- Programmable Interface Device-8155 I/O and Time
- Keyboard/Display Interface-8279
- General-Purpose Programmable Peripheral Devices-
  - Programmable Peripheral Interface-8255
  - Programmable Interval Timer-8254
  - Programmable Interrupt Controller-8259
  - DMA Controler-8237

Here in this project we are interfacing the 8255 Programmable Peripheral Interface to display the counter value from 00 to 99 second. The interfacing is done through a decoder/driver and the value is displayed on the 7-segment display. The program is used for generating the counting sequence and to control the displaying of the counting on the 7-segment display. The circuit is constructed as required and checked before connections are made with the 8085.

8085 kit:

The 8085 kit used is the DYNA-85. The MICROFRIEND DYNA-85 is a low cost learning and developing system for beginners as well as development engineers.

For the DYNA-85 kit the users' memory is from C000H to FFFFH. The monitor program uses memory from 0000H to 3FFFH. The I/O mapping for 8255 is:

10H-R/W of PORT A
11H- R/W of PORT B
12H- R/W of PORT C
13H- Write Control Register

Power supply is DMS SMPS-01

To enter the opcodes of the assembly language program the steps are:-
RESET → SET → <STARTING ADDRESS> → INR → <ENTER OPCODES> → INR → 76(HALT) → INR

To enter the data required at a particular location or starting from a location:-
RESET → SET → <STARTING ADDRESS> → INR → <ENTER DATA> → INR → <LAST DATA> → INR

To execute the entered program the steps are:-
RESET → GO → <STARTING ADDRESS> → EXEC

Elements required are:-

- 8085 Microprocessor kit
- 8255 Interface box
- A5611 common anode 7-segment display
- 74LS47 Decoder/driver
- 16 220ohm resistances
- Power supply
- Interface cable
- Jumpers or wires
- Bread board

About the 8085 microprocessor chip:-

It ha six general purpose registers of 8-bits each; identified as B,C,D,E,H and L. They can be combined as register pairs BC,DE and HL to perform 16-bit operations. A special register called the Accumulator is present which is used to perform arithmetic and logical operations. Also the ALU of the 8085 has five flip-flops which are set or reset depending on the contents of the accumulator register. The are Zero(Z), Carry(C), Sign(S), Parity(P) and Auxiliary Carry (AC). The most commonly used flags Zero, Carry and the Sign flags.

It is an 8-bit general purpose microprocessor capable of addressing 64K bytes of memory. The 8085 can operate at the maximum frequency of 5-MHz.

All the signal of 8085 can be classified into six groups:

- address bus
- data bus
- control and status signal
- power supply and frequency signal
- externally initiated signal
- serial I/O ports

Address bus: It has 16 signal lines that are used as the address bus; however these lines are spilt into 2 segments: A_{15}-A_8 and AD_7-AD_0. The eight signals lines A_{15}-A_8 are unidirectional and used for the most significant bits, called the high order address, of a 16 bit address. The signal lines AD_7-AD_0 are used for a dual purpose.

Multiplexed Address/ Data bus: The signal AD_7-AD_0 is bidirectional. They serve a dual purpose. They are used as the lower order address bus as well as the data bus. In executing an instruction, during the earlier part of the cycle, these lines are used as the low order address bus. During the late part of the cycle, these lines are used as the data bus.

Control and Status signals: The signals are—

ALE:-Address Latch Enable. This signal is used primarily to latch the low order address from the multiplexed bus and generate a separate set of 8 address lines A_7-A_0.

RD:-Read. This is active low. This signal indicates that the selected I/O or memory device is to be read and data are available on the data bus.

WR:-Write. This is active low. This signal indicates that the data on the data bus are to be written into a selected memory or I/O location.

IO/M'-when this signal is low, memory operation is done and when high it indicates IO operation.

S_1 and S_0:-these signals are similar to IO/M., can identify various operations.

Externally Initiated Signals:

INTR: Interrupt Request.

INTA': active low. Interrupt Acknowledge.

RST 7.5, RST 6.5, RST 5.5: Restart Interrupts.

TRAP: non-maskable Interrupt.

HOLD: when activate indicates a peripheral such as a DMA controller is requesting the use of the address and data bus.

HLDA: Hold Acknowledge.

READY: used to delay the microprocessor Read or Write cycle until a slow responding peripheral is ready to send or accept data.

Power supply and clock frequency:

V_{cc}: +5V power supply

V_{ss}: Ground Reference

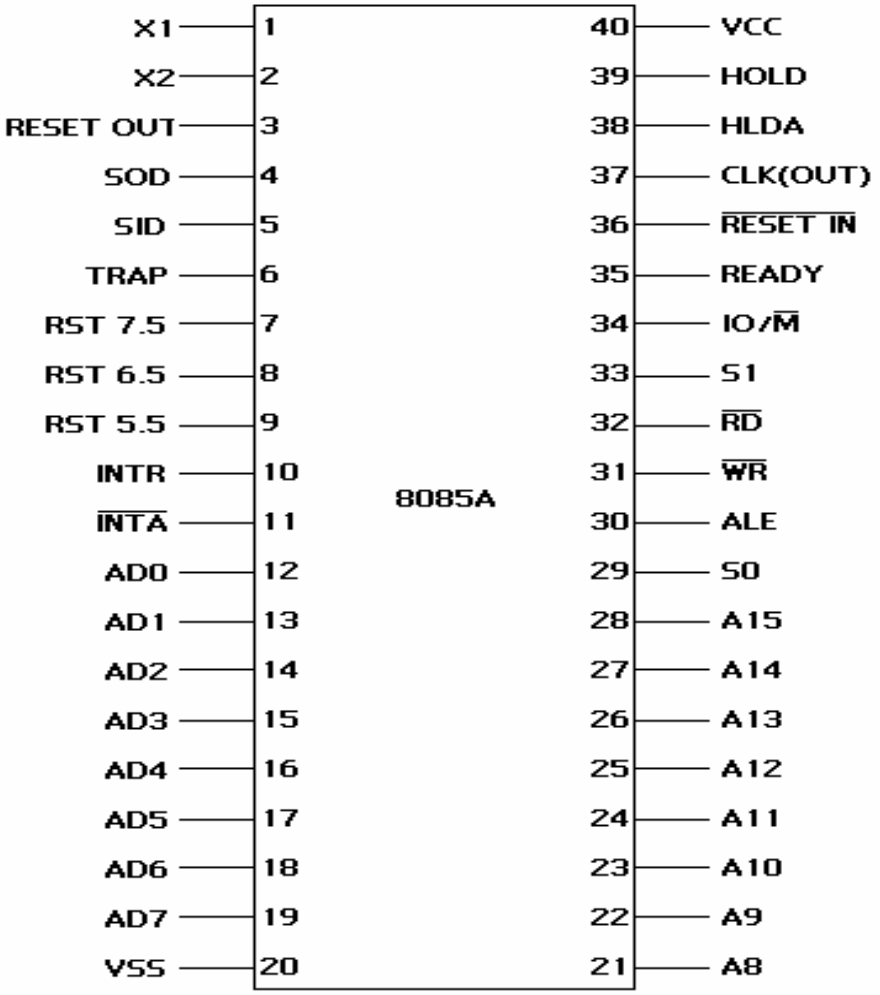
X_1,X_2: A crystal (or RC,LC network) is connected to these two pins. The frequency is internally divided by two; therefore, to operate a system at 3MHz, the crystal should have a frequency of 6MHz.

CLK-Clock Output: This signal can be used as the system clock for other devices.

RESET IN': active low. Used to set program counter zero.

RESET OUT**:** indicates that the MPU is being reset.

Serial I/O Ports: the 8085 has 2 signals to implement the serial transmission: SID (serial input data) and SOD (serial output data).



Pin Configuration of 8085A Microprocessor

Basic concept of how 8085 works:-

The 8085 works by decoding the instructions' opcode. Opcode is the representation of the instructions using 8-bit and represented in hex format which can be stored in the memory of the 8085 and in the registers of the 8085. While writing a program the program is written using the instructions then the opcode is written for the corresponding instructions. These opcodes are entered into the memory locations and the data on which the program will be performing the operations will be entered in the required memory locations. On execution of the program the microprocessor performs mainly three functions. They are:-

- Opcode Fetch
- Memory Read
- Memory Write

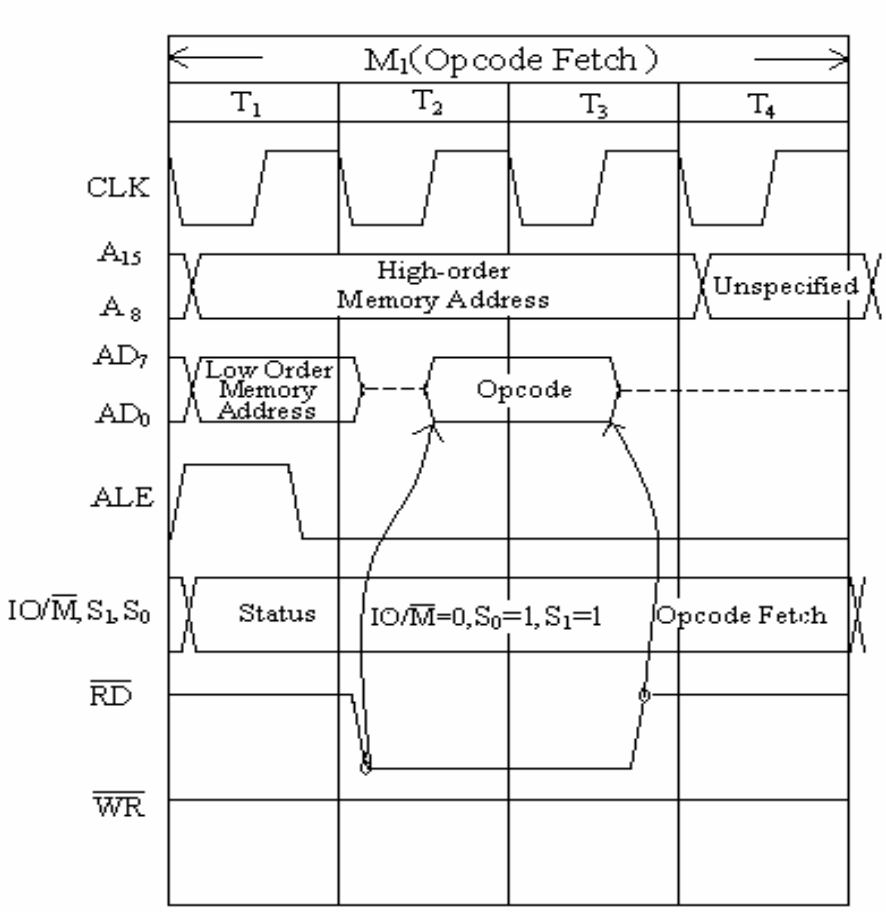
Opcode Fetch- It requires four T-States which is one Machine Cycle. It reads the opcode and then decodes it to perform the operation. The operation is performed using the address line and the control signals depending on the clock cycle.

Memory Read- It requires seven T-States and two Machine Cycle. In Machine Cycle 1 it fetches the opcode, decodes it and then in Machine Cycle 2 it reads the data byte from the memory location specified. The operation is performed using the address line and the control signals depending on the clock cycle.

Memory Write- It requires seven T-States and two Machine Cycle. In Machine Cycle 1 it fetches the opcode, decodes it and then in Machine Cycle 2 it writes the data byte from a register of the microprocessor to the memory location specified. The operation is performed using the address line and the control signals depending on the clock cycle.

Following is the description of the operations in detail with diagrams.

Opcode Fetch:-



Machine cycle for opcode fetch

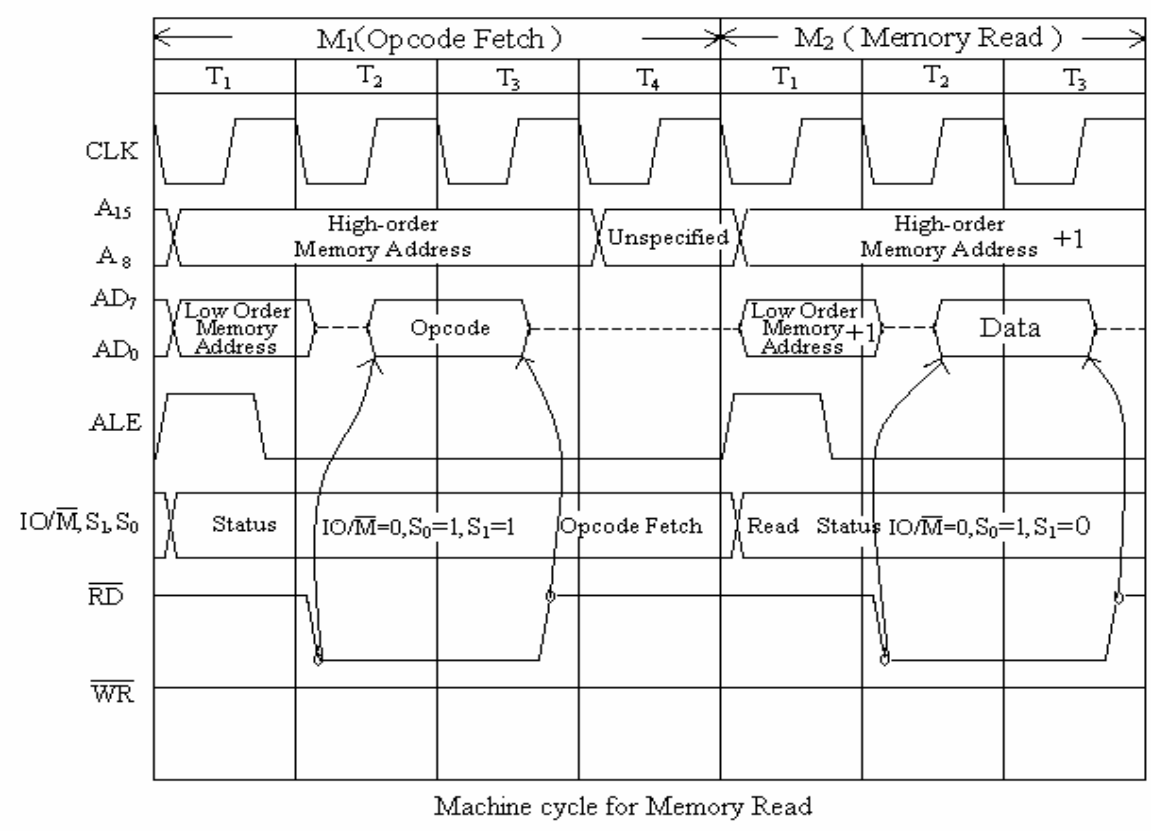
Step 1:- The microprocessor places the 16bit memory address from the program counter on the address bus. At T_1 the high order memory address is placed on the address lines A_{15}-$A_{8,}$ the low-order memory address is placed on the AD_7-AD_0, and the ALE signal goes high. Similarly, the status signal IO/M' goes low, indicating that this is a memory-related operation.

Step 2:- The control unit sends the control signal RD' to enable the memory chip. The control signal RD' is sent out during the clock period T_2, thus enabling the memory chip. The RD' signal is active during two clock periods.

Step 3:- The byte from the memory location is placed on the data bus. When the memory is enabled, the instruction byte is placed on the bus AD_7-AD_0 and transferred to the microprocessor. The RD' signal causes the instruction byte to be placed on bus AD_7-AD_0, and when RD' goes high, it causes the bus to go into high impedence.

Step 4:- The byte is placed in the instruction decoder of the microprocessor, and the task is carried out according to the instruction. The machine code or the byte is decoded by the instruction decoder, and the contents of the accumulator are copied into register C. this task is performed during the period T_4.

Memory Read:-

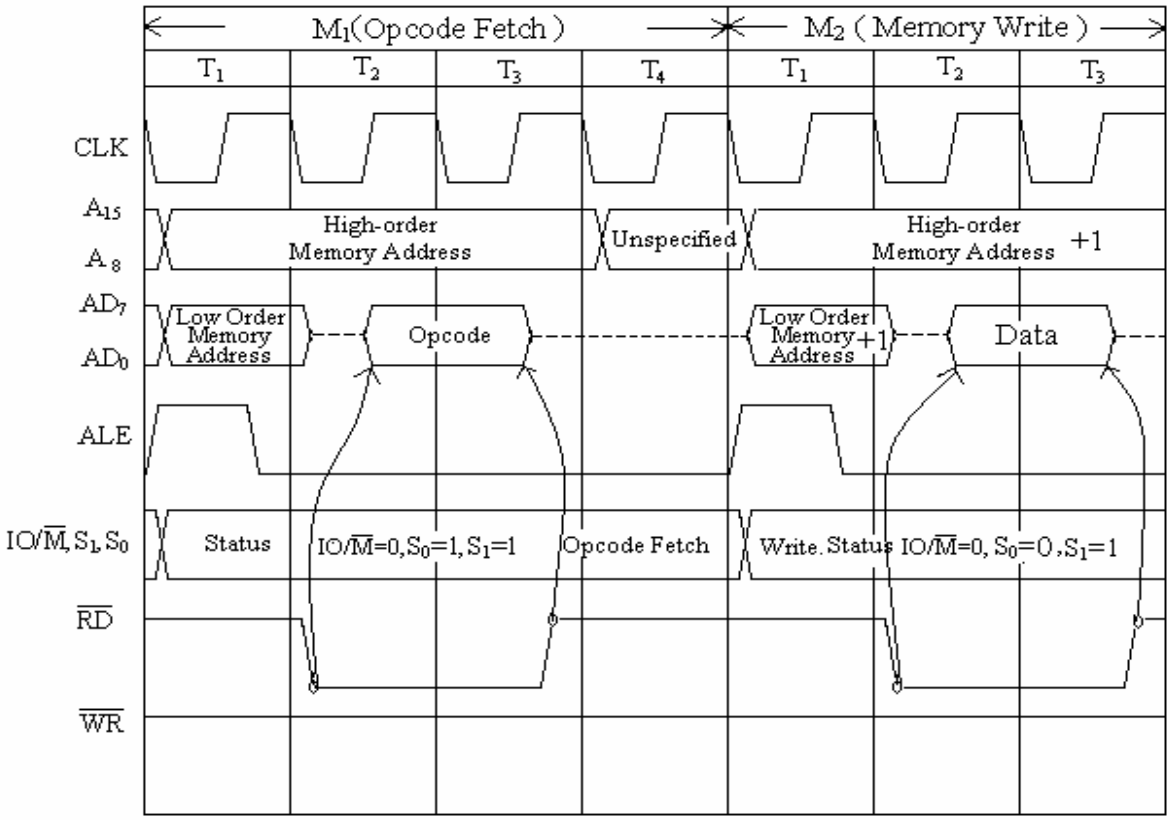


Machine cycle for Memory Read

Step 1:- The first machine cycle M_1 is Opcode Fetch. In this step it places the high order memory address of the program counter on the address lines A_{15}-A_8 of the address bus and low order memory address on the address lines AD_7-AD_0 of the address bus. Also the program counter is incremented to point the next machine location. Also at T_1 the microprocessor place's 011 on the status signal which is to identify that it is an opcode fetch cycle. The ALE signal goes high during T_1, which is used to latch the low order address from the address bus to the address lines AD_7-AD_0. The 8085 activates the RD' control signal, which enables the memory for the read operation of the opcode, and the memory places the opcode from the memory location pointed by the program counter on the data bus. Then the 8085 places the opcode in the instruction register and disables the RD' signal. The fetch cycle is completed in state T_3. During T_4, the 8085 decodes the opcode and if it finds out that a second bytes needs to be read then it performs Step 2. After the T_3 state, the contents of the bus A_{15}-A_8 are unknown and the data bus AD_7-AD_0 goes into high impedance state.

Step 2:- The second machine cycle M_2 is Memory Read cycle. At T_1 the ALE signal goes high which latches the lines AD_0-AD_7 to the lower order address of the of the program counter and place's 010 on the status signal. At T_2 the RD' signal becomes active and enables the memory chip for the read operation. At the rising edge of T_2 the 8085 activates the data bus as an input bus, memory places the data byte on the data bus, and the 8085 reads and stores the bytes in the accumulator during T_3.

Memory Write:-



Machine cycle for Memory Write

Step 1:- The first machine cycle M_1 is Opcode Fetch. In this step it places the high order memory address of the program counter on the address lines A_{15}-A_8 of the address bus and low order memory address on the address lines AD_7-AD_0 of the address bus. Also the program counter is incremented to point the next machine location. Also at T_1 the microprocessor places 011 on the status signal which is to identify that it is an opcode fetch cycle. The ALE signal goes high during T_1, which is used to latch the low order address from the address bus to the address lines AD_7-AD_0. The 8085 activates the RD' control signal, which enables the memory for the read operation of the opcode, and the memory places the opcode from the memory location pointed by the program counter on the data bus. Then the 8085 places the opcode in the instruction register and disables the RD' signal. The fetch cycle is completed in state T_3. During T_4, the 8085 decodes the opcode and if it finds out that a second bytes needs to be read then it performs Step 2. After the T_3 state, the contents of the bus A_{15}-A_8 are unknown and the data bus AD_7-AD_0 goes into high impedance state.

Step 2:- The second machine cycle M_2 is Memory Write cycle. At T_1 the ALE signal goes high which latches the lines AD_0-AD_7 to the lower order address of the of the program counter and place's 001 on the status signal. At T_2 the WR' signal becomes active and places the contents of the accumulator on the data bus AD_7-AD_0. During the T_3, the contents of the data bus are placed in memory location specified in the instruction.

About the 8255 peripheral chip:-

The 8255 is a widely used programmable chip, parallel I/O device. It can be programmed to transfer data under various conditions, from simple I/O to interrupt I/O. It is an important general purpose I/O device that can be used with almost any microprocessor.

8255 classified according to two modes:

- The bit set/ reset (BSR) mode
- The I/O mode.

BSR Mode:

The BSR mode is used to set or reset the bit in port C.

I/O Mode:

The I/O mode is further divided into 3modes:

Mode 0 - All ports functions as simple I/O ports.

Mode 1 - This is a handshake mode whereby port A and/or B use bits from port C as handshake signal.

Mode 2 - The port A can be set up for bidirectional data transfer using handshake signal from port C, and port B can be set up either in Mode 0 or in Mode 1.

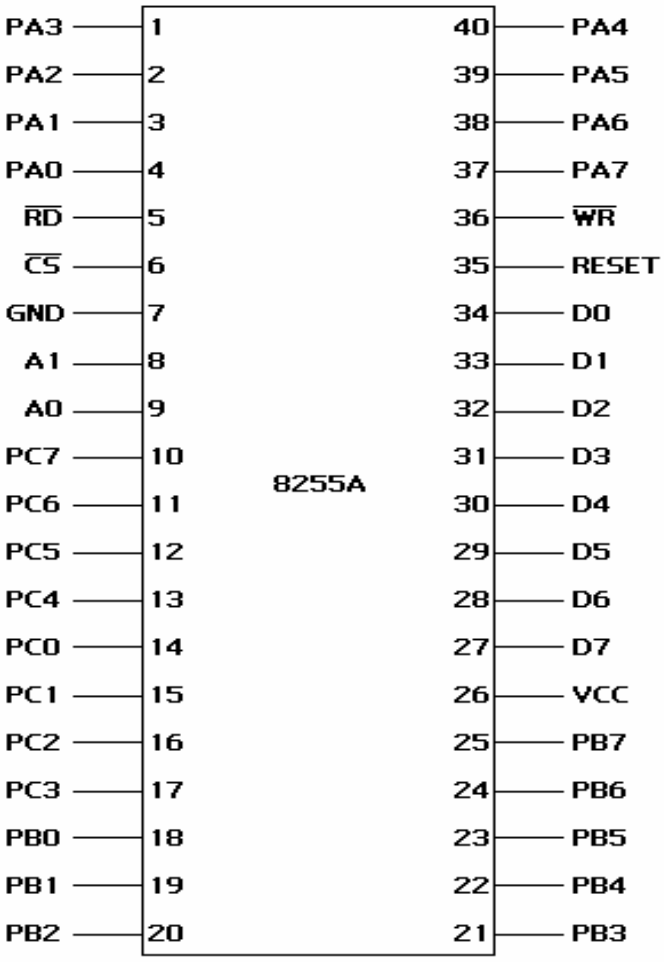
Control logic of the 8255 chip:-

RD' (read): this is active low. This control signal enables the Read operation. When the signal is low, the MPU reads data from a selected I/O port of the 8255.

WR' (write): this is active low. This control signal enables the Write operation. When the signal goes low, the MPU writes into a selected I/O port or the control register.

RESET (reset): this active high. It clears the control register and sets all the ports in the input mode.

CS',A_0 and A_1: CS is active low signal. These are device select signals. CS is connected to a decoded address, and A_0 and A_1 are generally connected to MPU address lines A_0 and A_1 respectively.



Pin Configuration of 8255A Programmable Peripheral Interface

The 7-segment display and the 74LS47 decoder/driver chip:-

The seven-segment LED display is a multiple display. It can display all decimal digits and some letters. It is very popular among multiple displays as it has the smallest number of separately controlled light emitting diodes [LED]. Multiple display of 9-segment LED, 14-segment LED and dot matrix type are available. These displays give better representation of alphanumeric characters but require complex circuitry.

In seven-segment display there are seven light emitting diodes [LED]. Each LED can be controlled separately. To display a digit or letter the desired segments are made ON. There are two types of 7-sement display namely, common-cathode type and common-anode type. In a common-cathode type display all the 7 cathodes of LEDs are tied together to the ground. When $+5V_{d.c}$ is applied to any segment, the corresponding diode light. Thus applying logic '1' i.e., positive logic to the desired segments, the desired letter or decimal number can be displayed. In a common-anode type play all the 7

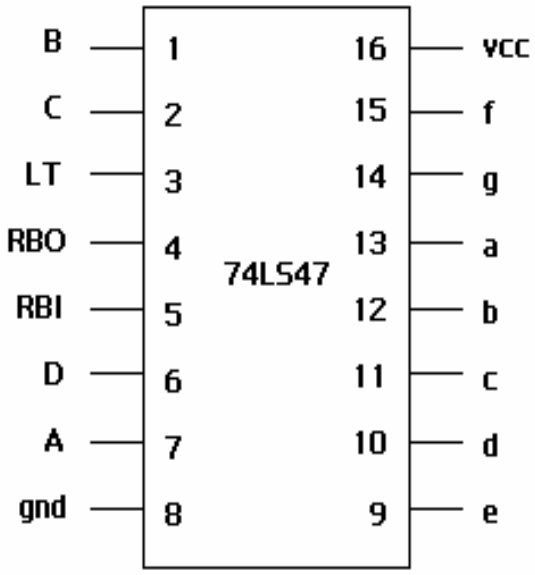
anodes are tied together and connected to +5V supply. A particular segment will emit light when 0 logic is applied to it.

The seven-segment displays are not connected to I/O ports directly. They are connected through buffers or drivers/ decoders. 7446A, 74L46, 7447, 74L47 and 74LS47 are decoders/ drivers for common-cathode type seven-segment displays.

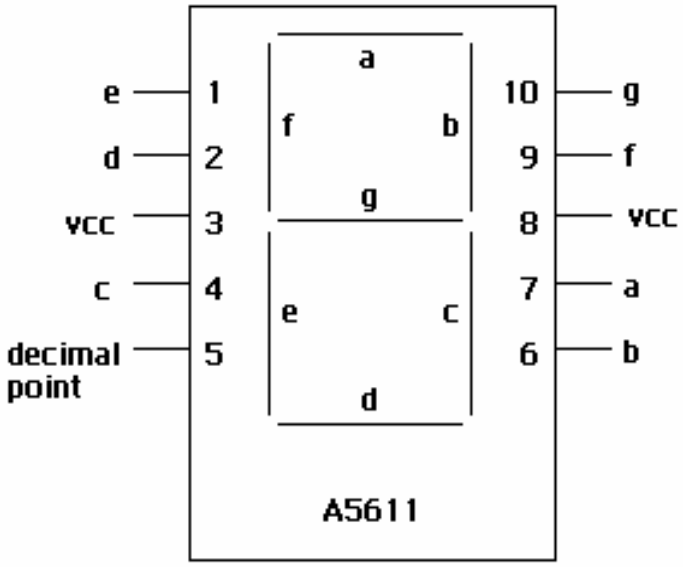
A5611

It is a common-anode 7-segment display.

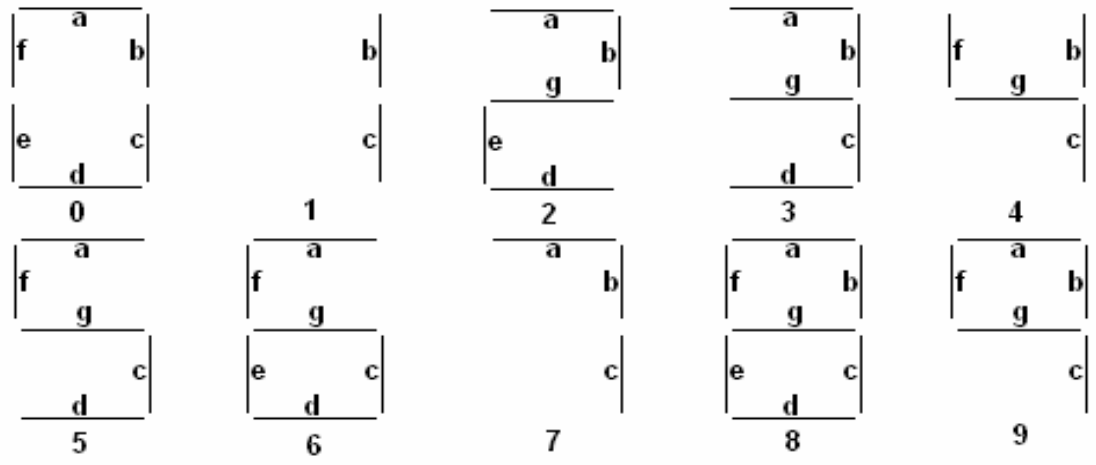
If pin diagram of 7-segment display is not known, it can be checked using multiplier. Set multimeter for ohm measurement. Connect its terminals to any two pins of 7-segment display.



Pin Diagram of 74LS47,
BCD to 7-Segment Decoder/Driver



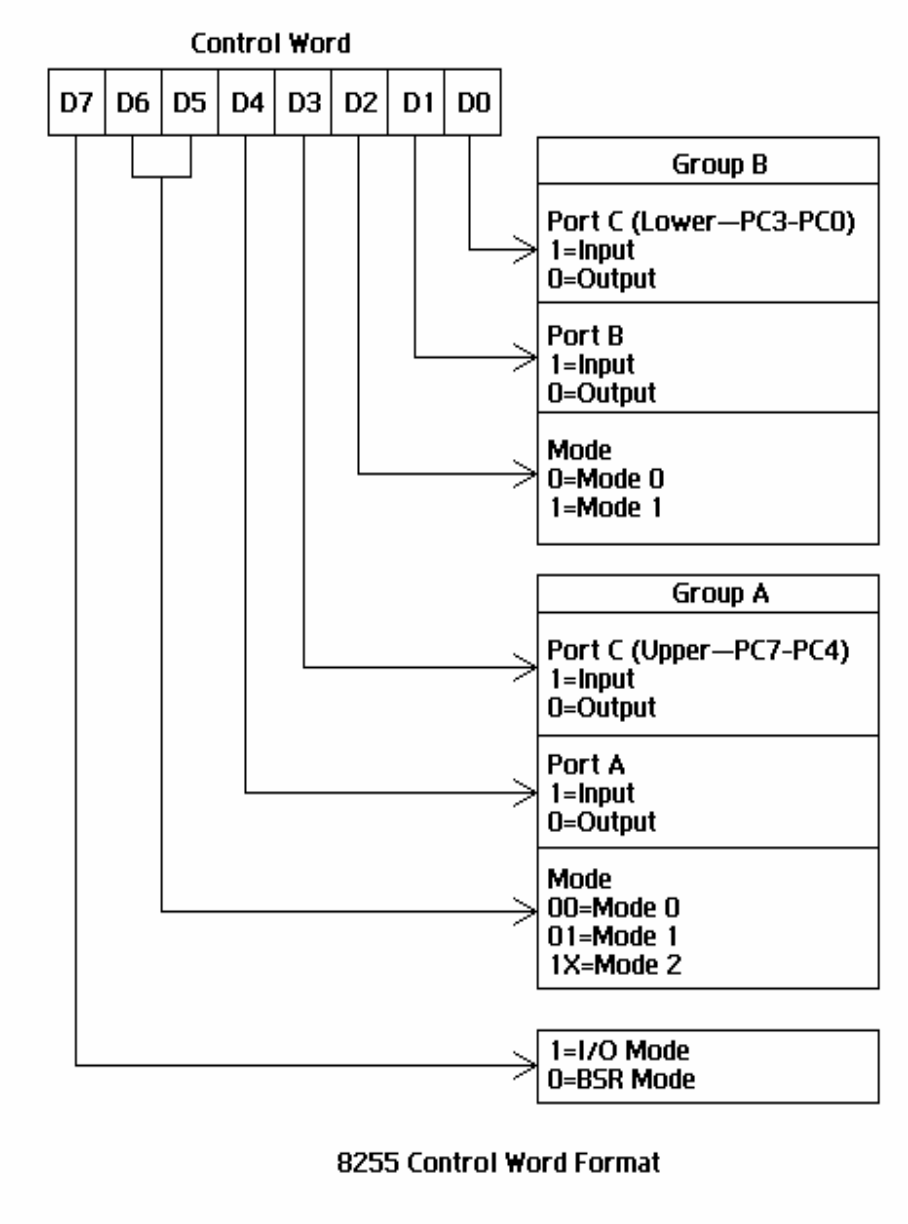
Pin Diagram of A5611,
Common Anode 7-Segment Display



Numerical designation for display on the seven segment

Designing the control word of the 8255:-

The control word 80H makes the port B an output port (given below is the representation of the control word of the 8255). The microprocessor outputs a hex number at the port B. The pins $PB_0 - PB_3$ of the port B are connected to the decoder/ driver 74LS47. Thus the binary bits corresponding to the decimal number of the lower order hex digit are applied to 74LS47 and it is displayed by the 7-segment display. The higher order hex digit's decimal number is displayed on the other 7-segment display. If the higher order digit is the output on the pins PB_4-PB_7 and these pins are not connected anywhere and consequently it is not displayed. If we use 2 decoder/ drivers and 7-segment displays, No.1 display will display LSB and No.2 display will show MSB. Thus two units of display will display 2 desired decimal numbers.

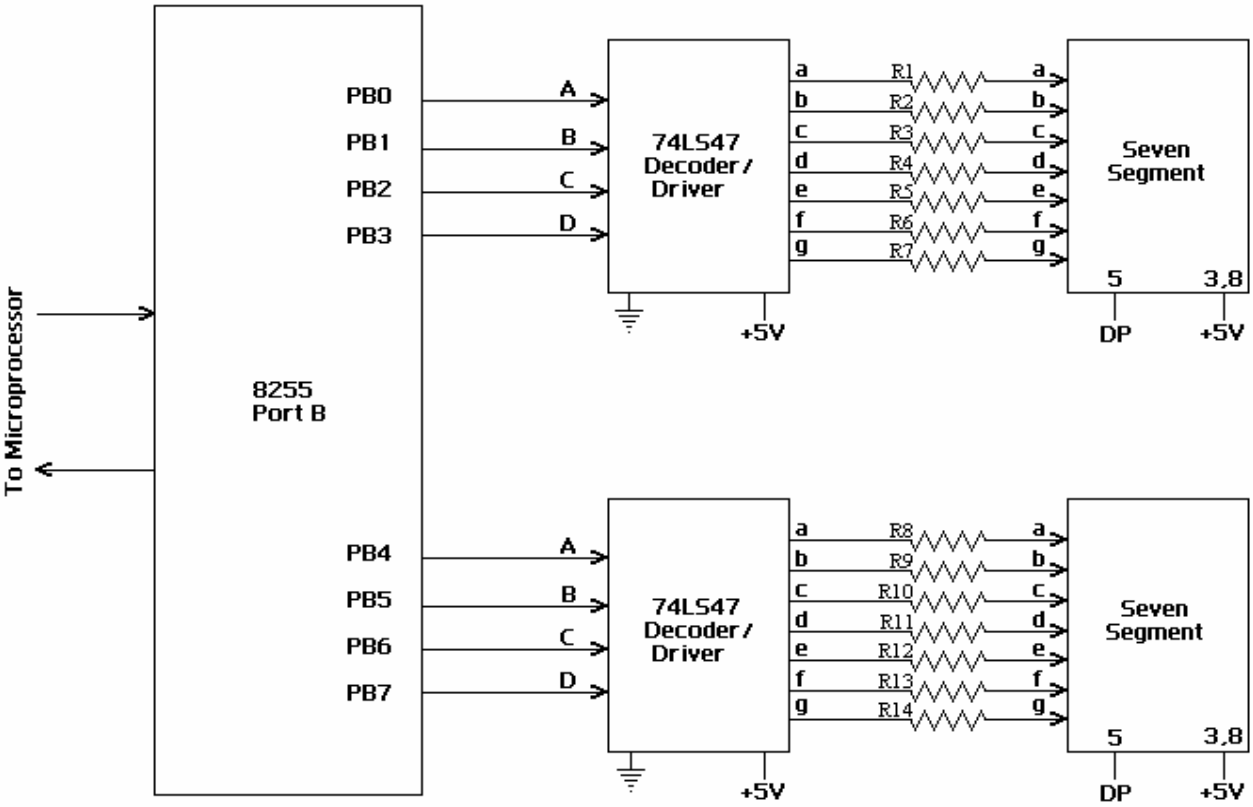


8255 Control Word Format

Interfacing the 8255 and the circuit for the decoder/driver and the 7-segment displays:-

Here we use two A5611 common anode 7-segment display and two 74LS47 BCD to 7-segment decoder/driver. As we want to display 1 to 99 counters, we have to use two 7-segment displays for the two digits and two decoder/driver for each 7-segment display. When the program will execute, the right most 7-segment display will display 0 to 9, nine times while the left most 7-segment display will display 0 to 9 maintaining the time delay during the other 7-segment display show 0 to 9. That means it will be incremented by 1, after each 0 to 9 counter of the right most 7-segment display. 7-segment display has 7 pins for the segments 'a' to 'g'. And other 3 pins for decimal point and vcc. The decoder/driver also has 7 pins corresponding to the 7-segment display (a to g), 4pins for input A,B,C and D and other for vcc, gnd etc. theses input pins are for the 4 bit binary number for representing 0 to 9.

We connect 7-segment display with decoder/driver in order 7-segment display's 'a' pin with decoder/driver's 'a' pin and so on via resistance. We use 8255 IO port. We join 4 input pins of each decoder/driver with the IO port PB_0-PB_7. PB_0 to PB_3 is connected to the input pin A to D of the 1st decoder/driver. PB_4 to PB_7 is connected to the input pin A to D of the 2nd decoder/driver.



R1-R14– Resistence of 220ohms

Interfacing two Seven-Segment Displays with 8255 Port B through 74LS47 Decoder/Driver

Algorithm

Step 1:- Initialize the stack pointer to the highest available user memory in the DYNA-85. The highest available memory location is FEFFH for the kit used.

Step 2:- Form the control word to access the ports of the 8255 peripheral chip. The D7 bit is 1 for I/O mode and all other bits are 0 which makes them output ports. Then write the control word to the control register of the 8255 whose address for the DYNA-85 kit used is 13H.

Step 3:- A counter is required so that at the end of the counting from 00H to 99H the counter must return to 00H again. So register E of the 8085 is used as that counter with an initial value of 02H.

Step 4:- Initialize register B as the main counter with an initial value of 00H which can count till 99 in decimal. Also this step is labeled as LBL so that at the end of the count the program control is returned to this instruction so that 00H is displayed.

Step 5:- Move the contents of register B to the accumulator A to display it on the seven-segment display via port B of the 8255 peripheral chip and the decoder driver chip. The address of port B of the 8255 is 11H for the DYNA-85 kit used.

Step 6:- Decrement the counter E by 1 and check if it is zero. If it is zero go to step 14 else continue from step 7.

Step 7:- Initialize register C with 00H which will be used for the counting loop termination.

Step 8:- Start the loop for counting and displaying the digits on the seven-segment display via port B of 8255 peripheral chip and the decoder driver chip. Move the contents of register B to accumulator A and display it on the seven-segment display.

Step 9:- Increment the value of the contents of the accumulator A, decimal adjust it using the DAA instruction and update it to the original counter stored in register B.

Step 10:- Call the delay subroutine.

- Step i:- In the delay subroutine push the contents of the accumulator and the status of the flags on top of the stack.
- Step ii:- Initialize register D with 01H which is just a dummy counter for making the delay to approximately 1second.
- Step iii:- Initialize register pair HL as a counter for delay of approximately 1 second with F424H.
- Step iv:- Decrement the counter in the register pair by 1.
- Step v:- Move the contents of the lower order to the accumulator and perform logical or with the higher order. If not zero then go to step iv else continue.
- Step vi:- Decrement the counter of register D by 1. If not zero then go to step iii else continue.
- Step vii:- Pop the contents of the accumulator and the status flags from the top of the stack to the accumulator and the flags.
- Step viii:- Return to the main program.

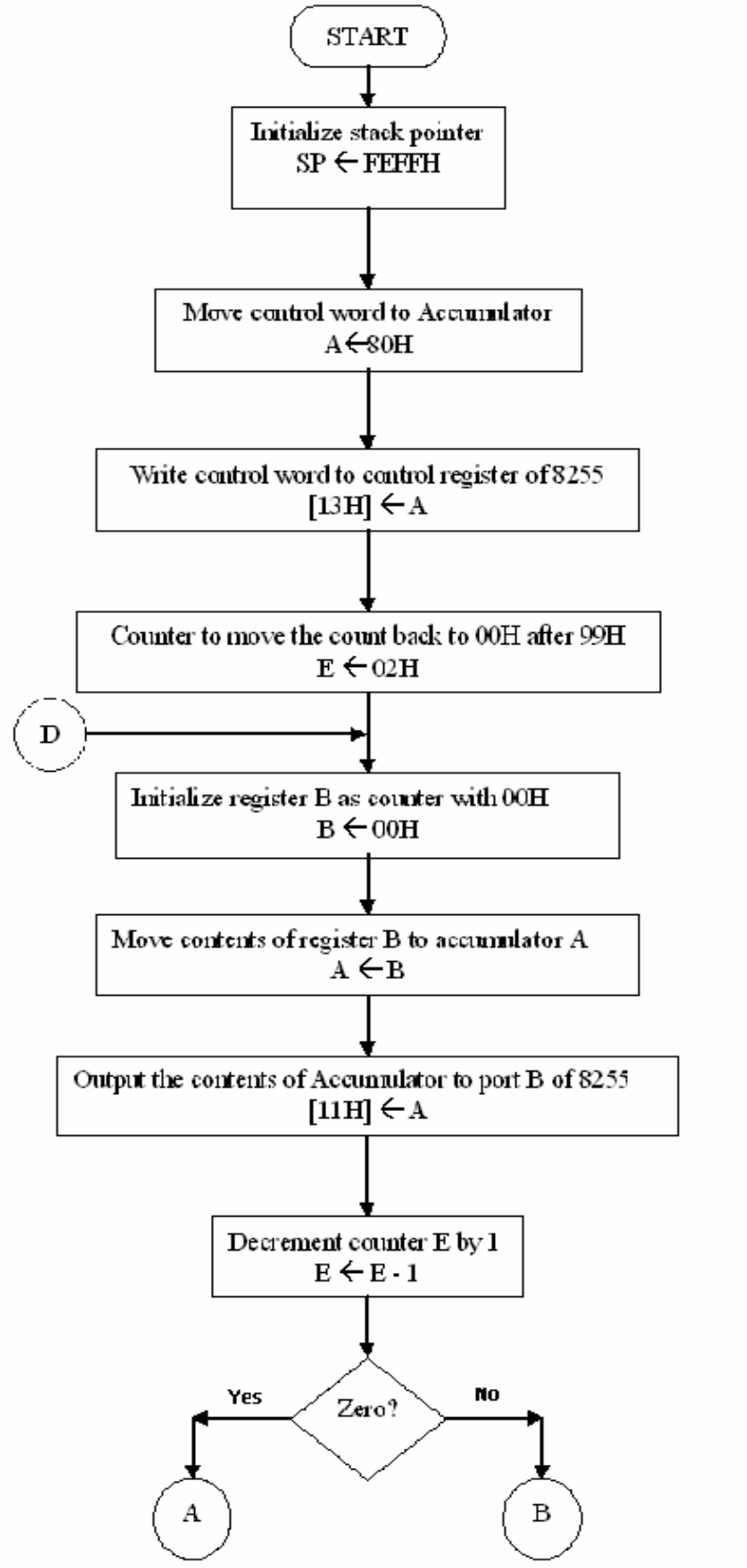
Step 11:- Move contents of register B to accumulator A and compare it with the contents of the register C.

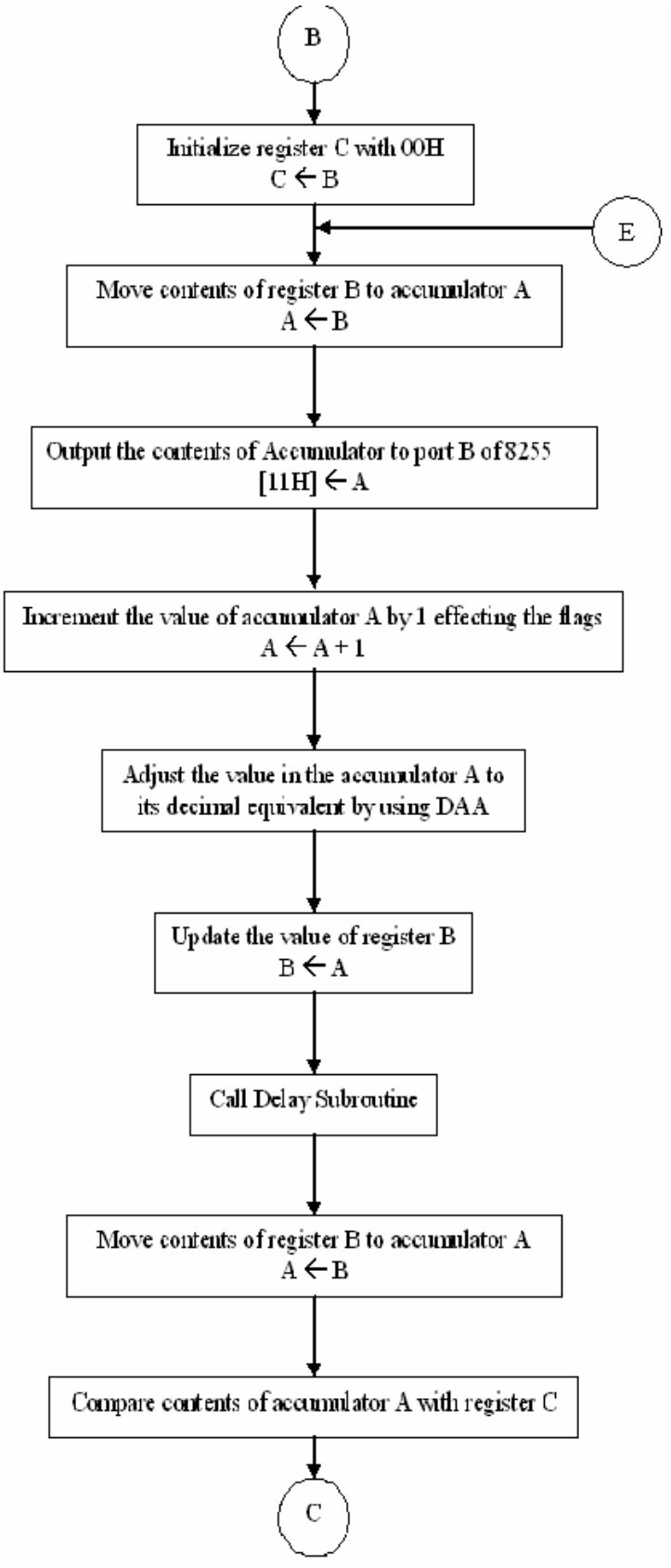
Step 12:- If not zero then go to step 8 else continue.

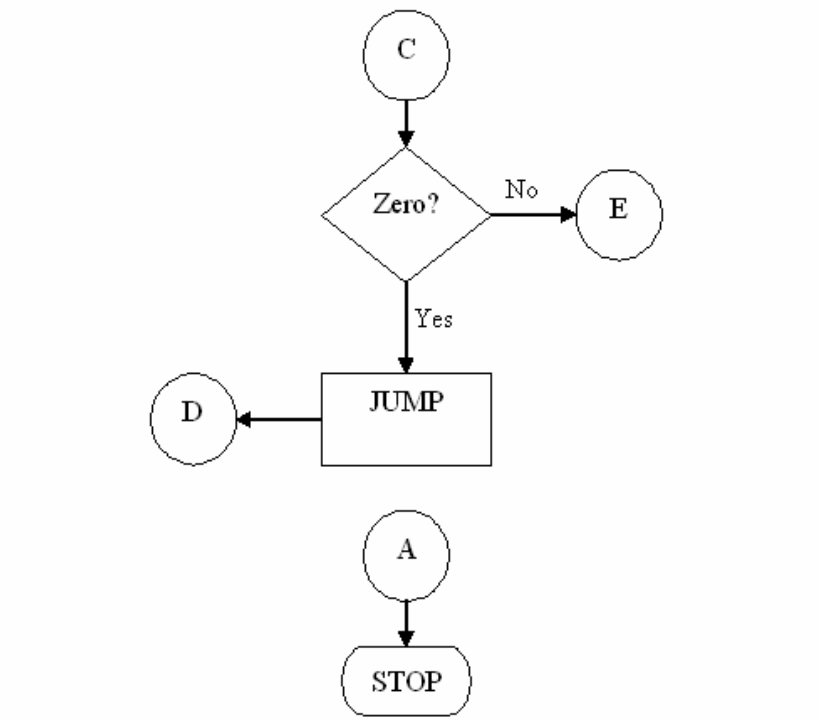
Step 13:- Jump unconditionally to step 4.

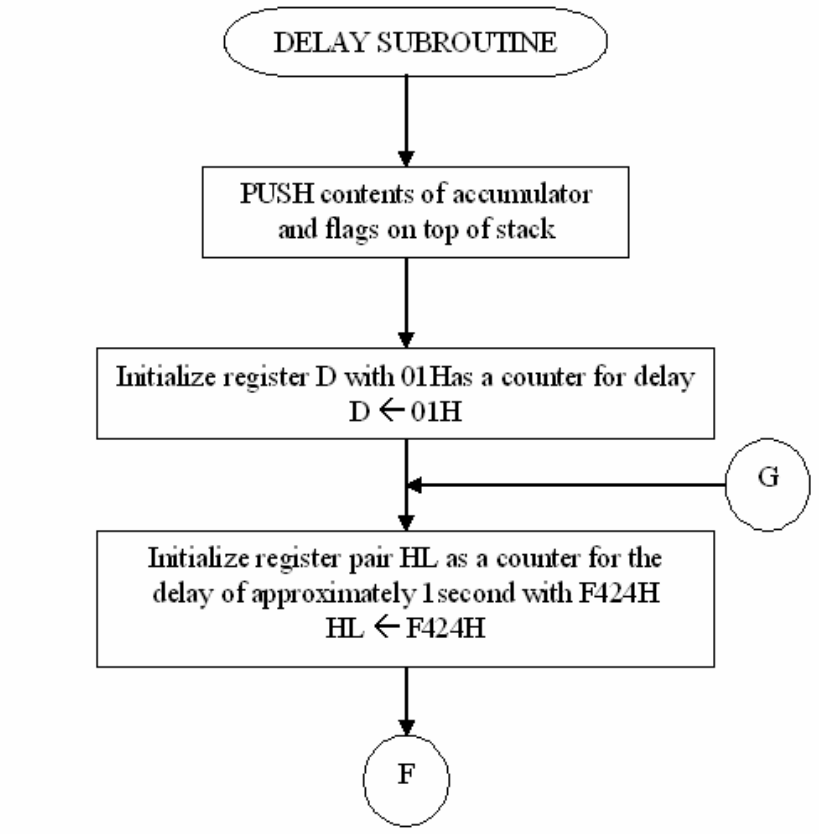
Step 14:- End of the program.

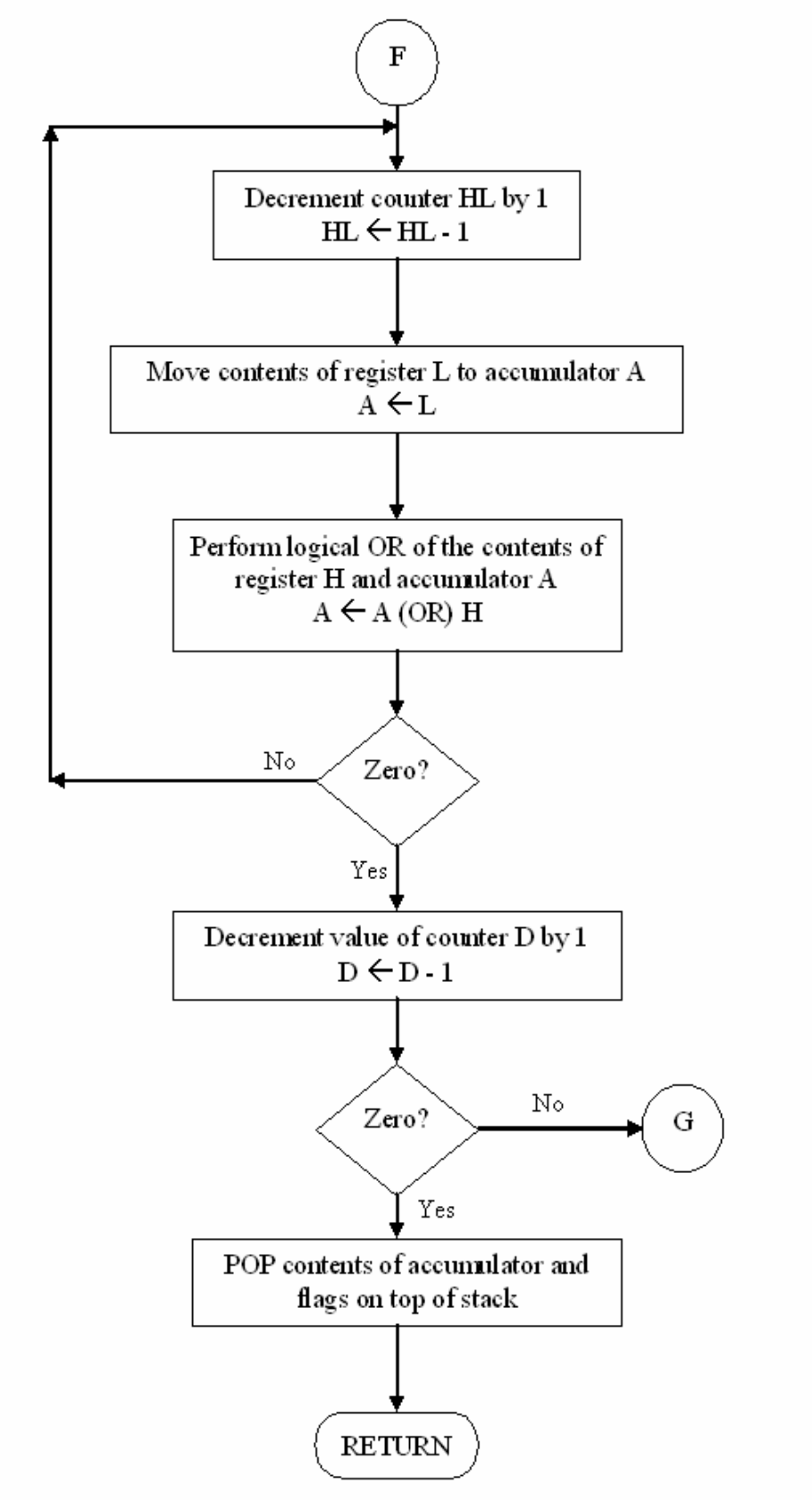
Flowchart


START
Initialize stack pointer
SP ← FEFFH
Move control word to Accumulator
A←80H
Write control word to control register of 8255
[13H] ←A
Counter to move the count back to 00H after 99H
E ←02H
D
Initialize register B as counter with 00H
B ←00H
Move contents of register B to accumulator A
A ←B
Output the contents of Accumulator to port B of 8255
[11H] ←A
Decrement counter E by 1
E ← E - 1
Zero?
Yes
No
A
B


B
Initialize register C with 00H
C ← B
E
Move contents of register B to accumulator A
A ← B
Output the contents of Accumulator to port B of 8255
[11H] ← A
Increment the value of accumulator A by 1 effecting the flags
A ← A + 1
Adjust the value in the accumulator A to
its decimal equivalent by using DAA
Update the value of register B
B ← A
Call Delay Subroutine
Move contents of register B to accumulator A
A ← B
Compare contents of accumulator A with register C
C


C
Zero?
No
E
Yes
JUMP
D
A
STOP


DELAY SUBROUTINE
PUSH contents of accumulator and flags on top of stack
Initialize register D with 01Has a counter for delay
D ← 01H
G
Initialize register pair HL as a counter for the delay of approximately 1second with F424H
HL ← F424H
F


F
Decrement counter HL by 1
HL ← HL - 1
Move contents of register L to accumulator A
A ← L
Perform logical OR of the contents of register H and accumulator A
A ← A (OR) H
Zero?
No
Yes
Decrement value of counter D by 1
D ← D - 1
Zero?
No
G
Yes
POP contents of accumulator and flags on top of stack
RETURN

Program with comments, mnemonics, address and opcode

Address	Opcode	Label	Mnemonics	Comments
F000H	31,FF,FE		LXI SP,FEFFH	Initialize Stack Pointer;SP <- FEFFH
F003H	3E,80		MVI A,80H	Move control word to Accumulator;A <- 80H
F005H	D3,13		OUT 13H	Write control word to control register of 8255;[13H] <- A
F007H	1E,02		MVI E,02H	Counter to move count back to 00H after 99H;E <- 02H
F009H	06,00	LBL:	MVI B,00H	Initialize register B as counter with 00H;B <- 00H
F00BH	78		MOV A,B	Move contents of register B to accumulator A;A <- B
F00CH	D3,11		OUT 11H	Output the contents of Accumulator to port B of 8255;[11H] <- A
F00EH	1D		DCR E	Decrement counter E by 1;E <- E - 1
F00FH	CA,25,F0		JZ STP	Check Zero flag.If Zero flag is set goto label STP else continue
F012H	48		MOV C,B	Initialize register C with 00H;C <- B
F013H	78	LOOP:	MOV A,B	Move contents of register B to accumulator A;A <- B
F014H	D3,11		OUT 11H	Output the contents of Accumulator to port B of 8255;[11H] <- A
F016H	C6,01		ADI 01H	Increment the value of accumulator A by 1 effecting flags;A <- A + 1
F018H	27		DAA	Adjust the value in the accumulator A to its decimal equivalent by using DAA
F019H	47		MOV B,A	Update the value of register B;B <- A
F01AH	CD,26,F0		CALL DELAY	Call Delay Subroutine
F01DH	78		MOV A,B	Move contents of register B to accumulator A;A <- B
F01EH	B9		CMP C	Compare contents of accumulator A with register C
F01FH	C2,13,F0		JNZ LOOP	Check Zero flag.If Zero flag is reset goto label LOOP else continue
F022H	C3,09,F0		JMP LBL	Jump unconditionally to label LBL
F025H	76	STP:	HLT	Stop
F026H	F5	DELAY:	PUSH PSW	PUSH contents of accumulator and flags on top of stack
F027H	16,01		MVI D,01H	Initialize register D with 01H as a counter for delay;D <- 01H
F029H	21,24,F4	LOOP2:	LXI H,F424H	Initialize register pair HL as a counter for delay of approximately 1 second with F424H;HL <- F424H
F02CH	2B	LOOP1:	DCX H	Decrement counter HL by 1;HL <- HL - 1
F02DH	7D		MOV A,L	Move contents of register L to accumulator A;A <- L
F02EH	B4		ORA H	Perform logical OR of the contents of register H and accumulator A;A <- A (OR) H
F02FH	C2,2C,F0		JNZ LOOP1	Check Zero flag.If Zero flag is reset goto label LOOP1 else continue
F032H	15		DCR D	Decrement value of counter D by 1;D <- D - 1
F033H	C2,29,F0		JNZ LOOP2	Check Zero flag.If Zero flag is reset goto label LOOP2 else continue
F036H	F1		POP PSW	POP contents of accumulator and flags on top of stack
F037H	C9		RET	Return

Discussions:-

Although it is said to be a counter of 0 to 99 seconds the timing is not accurate. Since the counter value of F424H is just an approximate calculation for a delay of 1 second. To make the delay of exactly 1 second we need to use a crystal.

This counter gives a count of 0 to 99 seconds when interfaced with only one I/O port of the 8255.Also the counter can be expanded from 0 to 99 seconds to 0 to 9999 seconds or 0 to 999999 seconds by interfacing two I/O ports of 8255 or all the three I/O ports of the 8255.

This counter can be used for different purposes like for the delay used in traffic signals where the signal lights are to be changed at an interval of 99 seconds and in other cases where a delay of 99 second is required.

While making the connections precautions must be taken. While connecting the interface cable the proper jumper must be connected. While making the circuit for the seven-segment display and the decoder proper pins must be connected and the wires must be cut short so that they don’t short circuit.

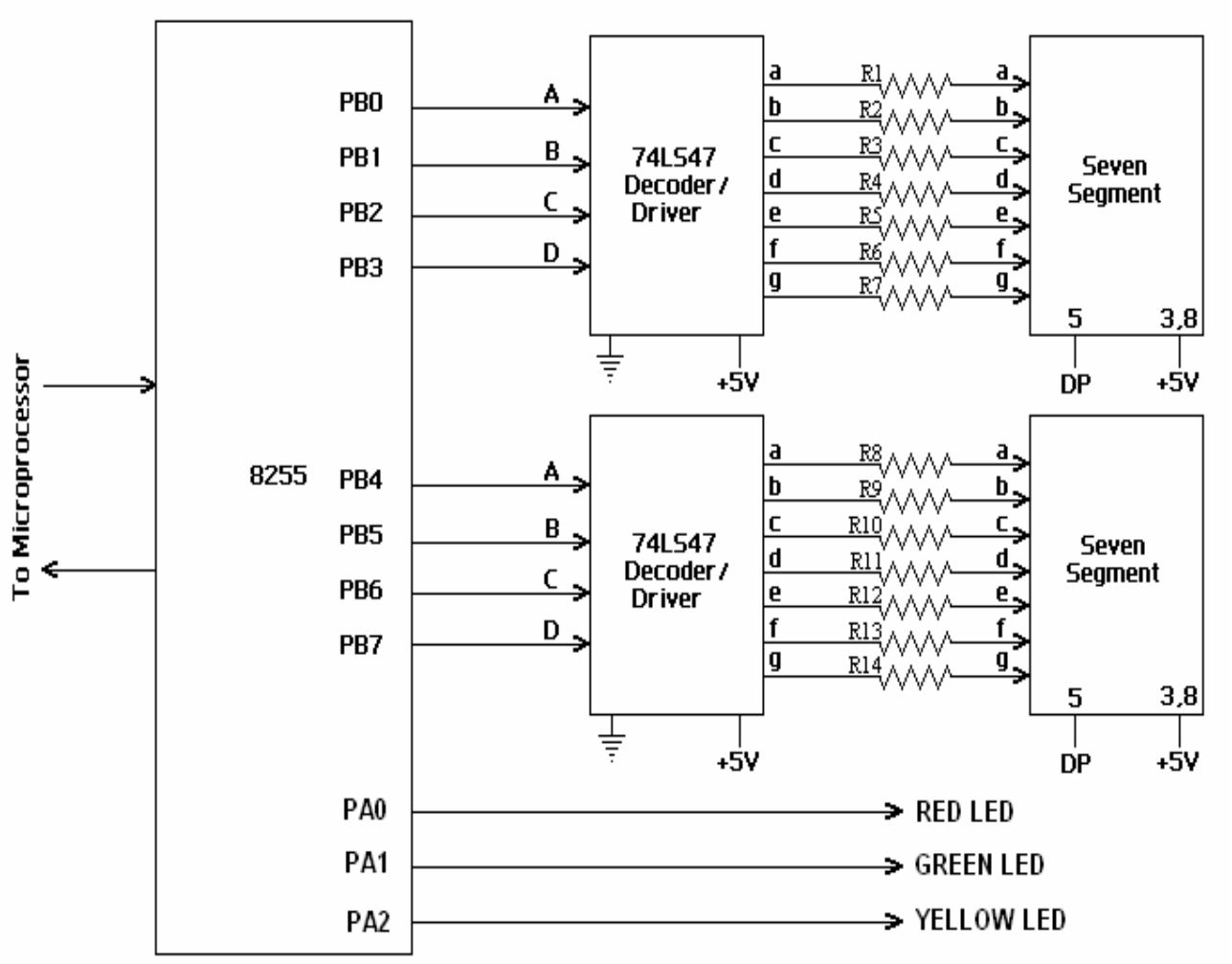
While entering the opcodes of the program in the microprocessor kit it must be done carefully otherwise the result will be wrong.

An application of this counter from 0 to 99 second:-

Traffic Signal

The counter has been used for designing a automated traffic signal. This prototype of the traffic signal lights up the red light then counts from 0 to 99 seconds and the lights up the green light and again counts from 0 to 99 seconds and finally lights up the yellow light and give a delay of 4 seconds approximately and again lights the red light and so on until the system is reset again.

The algorithm, program along with the mnemonics, opcodes , address and comments and a diagram of the circuit is given as follows.



R1-R14– Resistence of 220ohms

Interfacing two seven-Segment Displays with 8255 Port B through Decoder/Driver and connecting three LEDs with Port A of the 8255 for the Traffic Light signals

Algorithm

Main Program:-

Step I:- Initialize the stack pointer to the highest available user memory in the DYNA-85. The highest available memory location is FEFFH for the kit used.
Step II:- Form the control word to access the ports of the 8255 peripheral chip. The D7 bit is 1 for I/O mode and all other bits are 0 which makes them output ports. Then write the control word to the control register of the 8255 whose address for the DYNA-85 kit used is 13H.
Step III:- Move 01H to the accumulator for activating the red light and output it to the port A of 8255 peripheral chip whose address is 10H for DYNA-85 kit used. Then call the Count subroutine.
Step IV:- Move 02H to the accumulator for activating the green light and output it to the port A of 8255 peripheral chip whose address is 10H for DYNA-85 kit used. Then call the Count subroutine.
Step V:- Move 04H to the accumulator for activating the yellow light and output it to the port A of 8255 peripheral chip whose address is 10H for DYNA-85 kit used. Then call the delay subroutine four times for a delay of approximately 4 seconds.
Step VI:- Go to Step III until the system is reset.
Step VI:- End of the program

Count Subroutine:-

Step 1:- In the count subroutine push the contents of the accumulator and the status of the flags on top of the stack.
Step 2:- A counter is required so that at the end of the counting from 00H to 99H the counter must return to 00H again. So register E of the 8085 is used as that counter with an initial value of 02H.
Step 3:- Initialize register B as the main counter with an initial value of 00H which can count till 99 in decimal. Also this step is labeled as LBL so that at the end of the count the program control is returned to this instruction so that 00H is displayed.
Step 4:- Move the contents of register B to the accumulator A to display it on the seven-segment display via port B of the 8255 peripheral chip and the decoder driver chip. The address of port B of the 8255 is 11H for the DYNA-85 kit used.
Step 5:- Decrement the counter E by 1 and check if it is zero. If it is zero go to step 14 else continue from step 7.
Step 6:- Initialize register C with 00H which will be used for the counting loop termination.
Step 7:- Start the loop for counting and displaying the digits on the seven-segment display via port B of 8255 peripheral chip and the decoder driver chip. Move the contents of register B to accumulator A and display it on the seven-segment display.
Step 8:- Increment the value of the contents of the accumulator A, decimal adjust it using the DAA instruction and update it to the original counter stored in register B.
Step 9:- Call the delay subroutine.

Step 10:- Move contents of register B to accumulator A and compare it with the contents of the register C.
Step 11:- If not zero then go to step 7 else continue.
Step 12:- Jump unconditionally to step 3.
Step 13:- Pop the contents of the accumulator and the status flags from the top of the stack to the accumulator and the flags.
Step 14:- Return to the part of the program from where it was called.

Delay Subroutine:-

Step i:- In the delay subroutine push the contents of the accumulator and the status of the flags on top of the stack.
Step ii:- Initialize register D with 01H which is just a dummy counter for making the delay to approximately 1second.
Step iii:- Initialize register pair HL as a counter for delay of approximately 1 second with F424H.
Step iv:- Decrement the counter in the register pair by 1.
Step v:- Move the contents of the lower order to the accumulator and perform logical or with the higher order. If not zero then go to step iv else continue.
Step vi:- Decrement the counter of register D by 1. If not zero then go to step iii else continue.
Step vii:- Pop the contents of the accumulator and the status flags from the top of the stack to the accumulator and the flags.
Step viii:- Return to the part of the program from where it was called.

Program with comments, mnemonics, address and opcode

Address	Opcode	Label	Mnemonics	Comments
F000H	31,FF,FE		LXI SP,FEFFH	Initialize Stack Pointer;SP <- FEFFH
F003H	3E,80		MVI A,80H	Move control word to Accumulator;A <- 80H
F005H	D3,13		OUT 13H	Write control word to control register of 8255;[13H] <- A
F007H	3E,01	LOOP3:	MVI A,01H	For activating the RED light
F009H	D3,10		OUT 10H	Write it to the port A of 8255 for red light;[10H] <- A
F00BH	CD,29,F0		CALL COUNT	Call Count subroutine for a delay of approximately 99 second
F00EH	3E,02		MVI A,02H	For activating the GREEN light
F010H	D3,10		OUT 10H	Write it to the port A of 8255 for green light;[10H] <- A
FO12H	CD,29,F0		CALL COUNT	Call Count subroutine for a delay of approximately 99 second
F015H	3E,04		MVI A,04H	For activating the YELLOW light
F017H	D3,10		OUT 10H	Write it to the port A of 8255 for yellow light;[10H] <- A
F019H	CD,4A,F0		CALL DELAY	Call Delay subroutine for a delay of approximately 1 second
F01CH	CD,4A,F0		CALL DELAY	Call Delay subroutine for a delay of approximately 1 second
F01FH	CD,4A,F0		CALL DELAY	Call Delay subroutine for a delay of approximately 1 second
F022H	CD,4A,F0		CALL DELAY	Call Delay subroutine for a delay of approximately 1 second
F025H	C3,07,F0		JMP LOOP3	Jump unconditionally to label LOOP3
F028H	76		HLT	Stop
F029H	F5	COUNT:	PUSH PSW	PUSH contents of accumulator and flags on top of stack
F02AH	1E,02		MVI E,02H	Counter to move count back to 00H after 99H;E <- 02H
F02CH	06,00	LBL:	MVI B,00H	Initialize register B as counter with 00H;B <- 00H
F02EH	78		MOV A,B	Move contents of register B to accumulator A;A <- B
F02FH	D3,11		OUT 11H	Output the contents of Accumulator to port B of 8255; [11H] <- A
F031H	1D		DCR E	Decrement counter E by 1;E <- E - 1
F032H	CA,48,F0		JZ STP	Check Zero flag.If Zero flag is set goto label STP else continue
F035H	48		MOV C,B	Initialize register C with 00H;C <- B
F036H	78	LOOP:	MOV A,B	Move contents of register B to accumulator A;A <- B
F037H	D3,11		OUT 11H	Output the contents of Accumulator to port B of 8255;[11H] <- A

Address	Opcode	Label	Mnemonics	Comments
F039H	C6,01		ADI 01H	Increment the value of accumulator A by 1 effecting flags; A <- A + 1
F03BH	27		DAA	Adjust the value in the accumulator A to its decimal equivalent by using DAA
F03CH	47		MOV B,A	Update the value of register B;B <- A
F03DH	CD,4A,F0		CALL DELAY	Call Delay Subroutine
F040H	78		MOV A,B	Move contents of register B to accumulator A;A <- B
F041H	B9		CMP C	Compare contents of accumulator A with register C
F042H	C2,36,F0		JNZ LOOP	Check Zero flag.If Zero flag is reset goto label LOOP else continue
F045H	C3,2C,F0		JMP LBL	Jump unconditionally to label LBL
F048H	F1	STP:	POP PSW	POP contents of accumulator and flags from top of stack
F049H	C9		RET	Return
F04AH	F5	DELAY:	PUSH PSW	PUSH contents of accumulator and flags on top of stack
F04BH	16,01		MVI D,01H	Initialize register D with 01H as a counter for delay; D <- 01H
F04DH	21,24,F4	LOOP2:	LXI H,F424H	Initialize register pair HL as a counter for delay of approximately 1 second with F424H;HL <- F424H
F050H	2B	LOOP1:	DCX H	Decrement counter HL by 1;HL <- HL - 1
F051H	7D		MOV A,L	Move contents of register L to accumulator A;A <- L
F052H	B4		ORA H	Perform logical OR of the contents of register H and accumulator A;A <- A (OR) H
F053H	C2,50,F0		JNZ LOOP1	Check Zero flag.If Zero flag is reset goto label LOOP1 else continue
F056H	15		DCR D	Decrement value of counter D by 1;D <- D - 1
F057H	C2,4D,F0		JNZ LOOP2	Check Zero flag.If Zero flag is reset goto label LOOP2 else continue
F05AH	F1		POP PSW	POP contents of accumulator and flags from top of stack
F05BH	C9		RET	Return

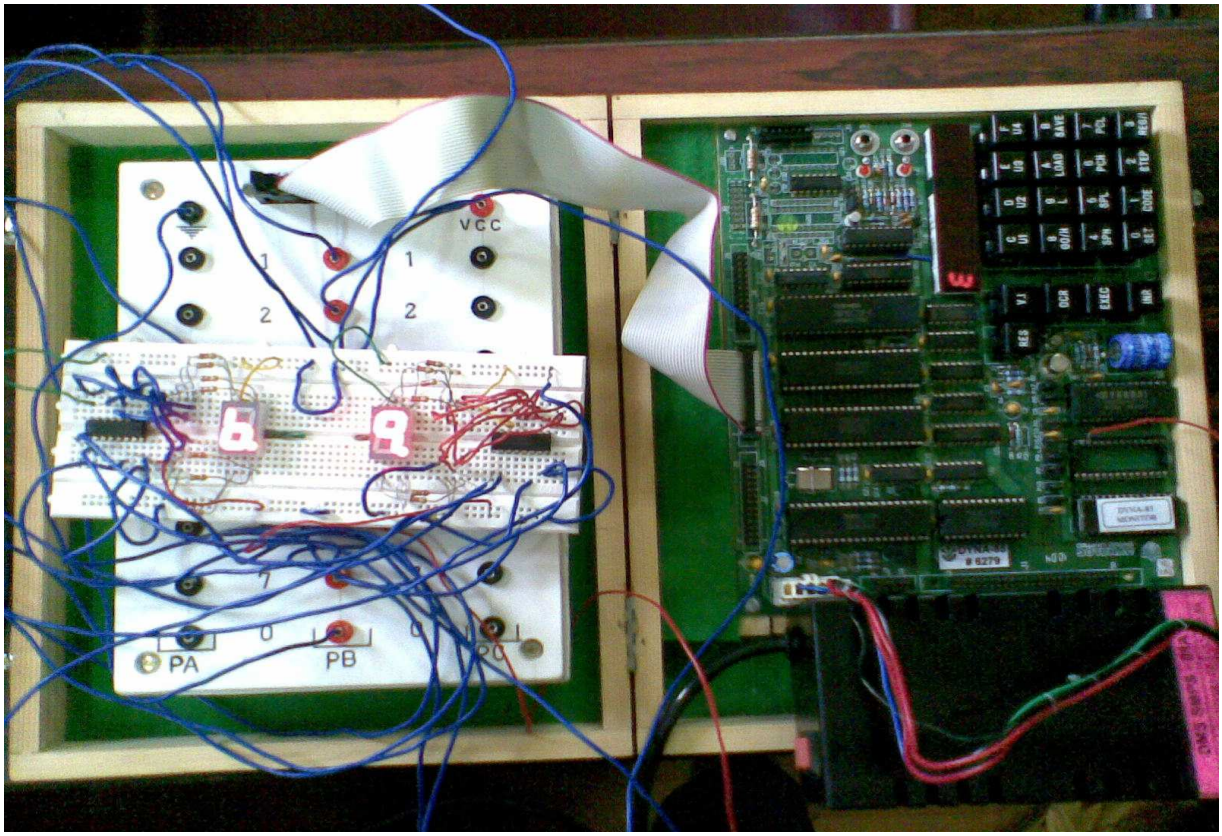
Conclusion

This project can also be done using 8086 microprocessor and other microprocessors after 8085. In these microprocessors the accuracy would be more precise than the one using 8085 but not as precise as it would be when done with a crystal. The crystal provides more accuracy than the counter for the time delay. The interfacing must be done correctly otherwise the kit or the chips may be hampered. So care must be taken.

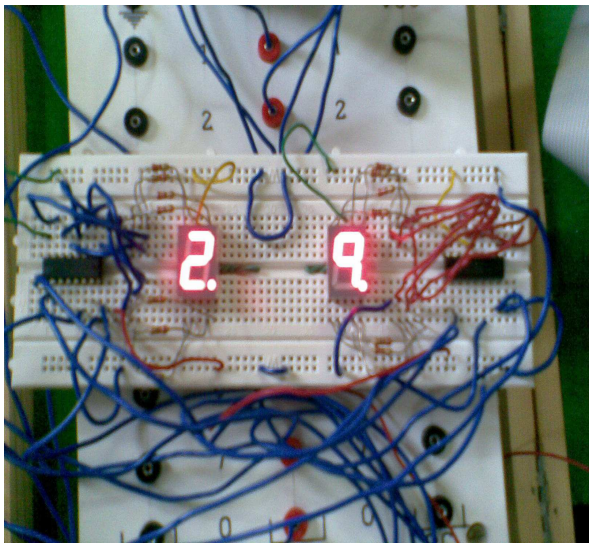
Reference

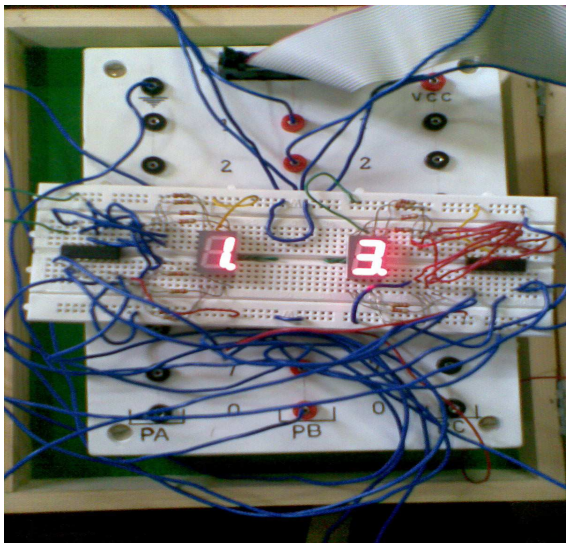
- Ramesh Gaonkar, State University of New York, O.C.C Campus at Syracuse - *Microprocessor Architecture, Programming and Application with the 8085*; Fifth Edition; Penram International Publishing (India) Private Limited
- B.Ram, Professor and head(Retd.) Electrical Engineering and Dean of Faculty of engineering Patna University, Patna – *Fundamentals of Microprocessor and Microcomputer*; Sixth Edition; Dhanpat Rai Publication

Images of the setup of the project



The full interfacing with the 8085-Dyna kit.





Displays of the counting sequence on the seven-segment displays.