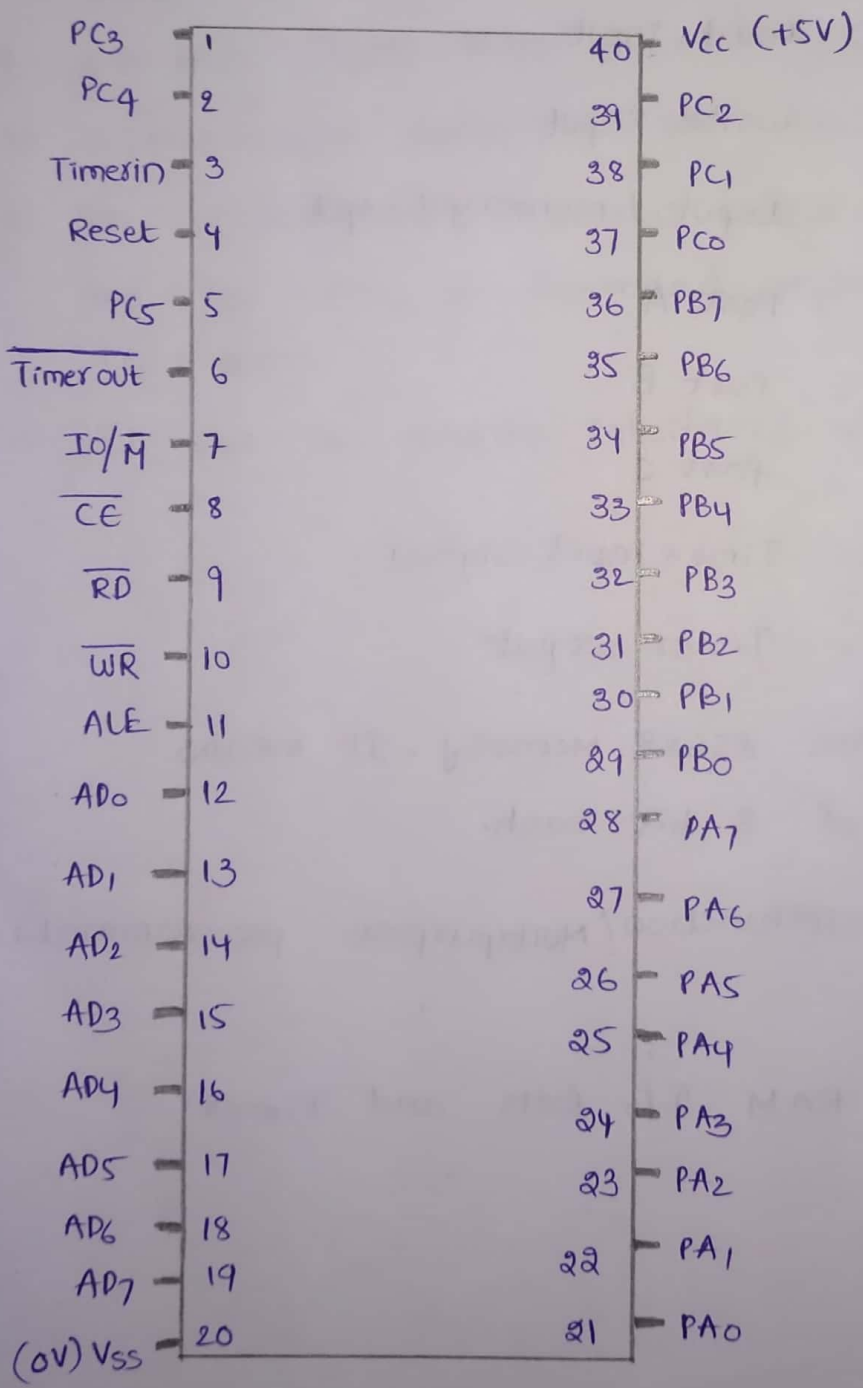
## Group 10

1. Draw neatly the pin diagram and architectural representation of a multipurpose programmable device 8155. Describe about its control logics, I/O ports. How can you determine the addresses for I/O ports, timer and control register?
2. Illustrate the interfacing of seven segment LED output ports using 8155.
3. Write a program to count from 0 to 20H with a delay of 100ms between each count. After the count 20H, the counter should reset itself and repeat the sequence. Use register pair DE as a delay register. Draw a flowchart and show your calculation to set up the 100ms delay.

# MICROPROCESSOR MINI PROJECT

1912149
CSE - B
NITIN SAI

1. Draw neatly the pin diagram and architectural representation of multipurpose programmable device 8155. Describe about its control logics, I/o ports. How can you determine the addresses for I/o ports, timer and control register ?




Pin diagram of 8155 processor (Intel corporation)

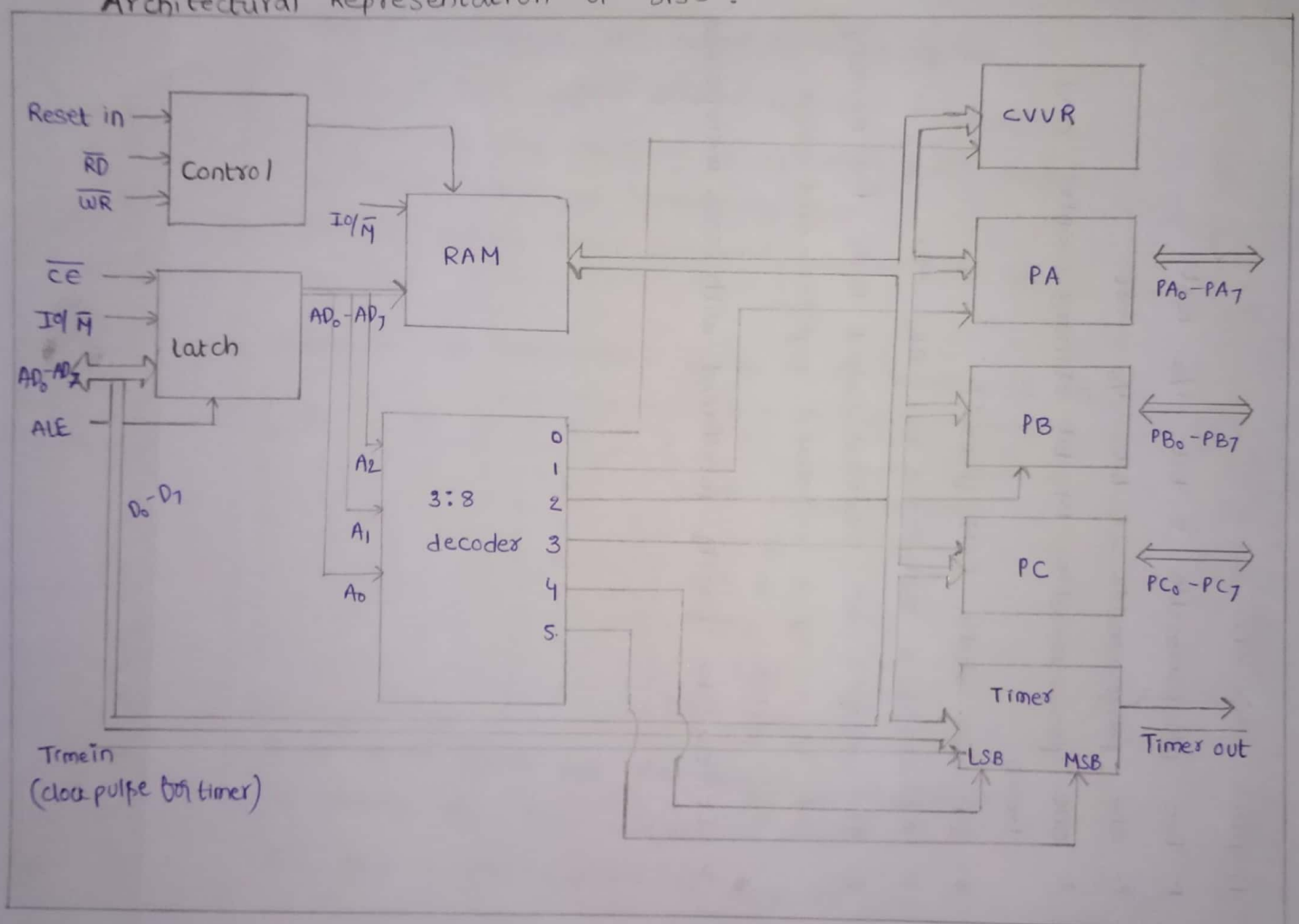
1. $AD_0 - AD_7$ : Address/Data bus multiplexed
2. RESET : RESET Input
3. $\overline{CE}$ : chip Enable
4. ALE : Arithmetic latch Enable
5. $\overline{RD}$ : Read input
6. $\overline{WR}$ : write input
7. $IO/\overline{M}$ : Input / memory / output
8. $PA_0 - PA_7$ : Port A
9. $PB_0 - PB_7$ : Port B
10. $PC_0 - PC_7$ : Port C
11. TIMER-IN : Timer input signal
12. $\overline{\text{TIMER-OUT}}$ : Timer output

- This section has 256X8 Memory. It means 256 locations of 8 bit each.
- 8155 is a multifunction/Multipurpose programmable device
- It contains RAM, I/o Ports and Timer.

Features of 8155 :

1. Two programmable 8 bit i/o Port
2. one programmable 6 bit i/o port
3. one programmable 14 bit binary counter and timer.
4. 2K bits static RAM (256 x 8)
5. Address/Data Multiplexed bus $AD_0 - AD_7$
6. It contains an internal select logic for memory and i/o using a command register and two i/o ports.
7. It can be easily interfaced with 8085 microprocessor.

Architectural Representation of 8155 :



| $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ |
|---|---|---|---|---|---|---|---|
| Timer Command | | $IE_B$ | $IE_A$ | PC(ALT$_1$ and ALT$_0$) | | $P_B$ | $P_A$ |

Control register is also called Command register

- It is a 8-bit which defines different functions.
- It defines the function of different ports such as input, output or handshaking mode.
- It also defines timer command for timer to start and stop.

Based on the combinations of $A_0$, $A_1$, $A_2$ selection is done.

| $A_2$ | $A_1$ | $A_0$ | |
|---|---|---|---|
| 0 | 0 | 0 | control/status register |
| 0 | 0 | 1 | Port A |
| 0 | 1 | 0 | Port B |
| 0 | 1 | 1 | Port C |
| 1 | 0 | 0 | LSB Timer |
| 1 | 0 | 1 | MSB Timer |

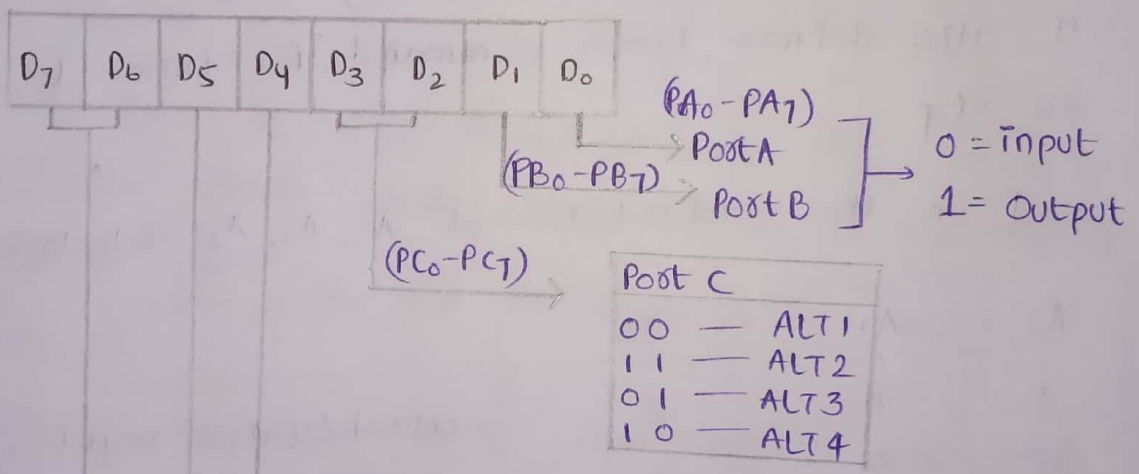
It is to be noted that the control/status register is having the same address (ie. 20H)

But the control register is accessed with $\overline{WR}=0$, $\overline{RD}=1$.

For status register access $\overline{WR}=1$ and $\overline{RD}=0$.

The control register can never be read

| $A_7$ | $A_6$ | $A_5$ | $A_4$ | $A_3$ | $A_2$ | $A_1$ | $A_0$ | Address | Register/Port/Timer |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 20H | Control/Status register |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 21H | Port A |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 22H | Port B |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 23H | Port C |
| 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 24H | LSB Timer |
| 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 25H | MSB Timer |




Timer Commands

00 = NOP → No effect on timer. ie. no effect on counter

01 = Stop → Stop counting if timer is running otherwise no effect on timer.

10 = Stop after terminal count, stop after end of count if timer is running. else no affect

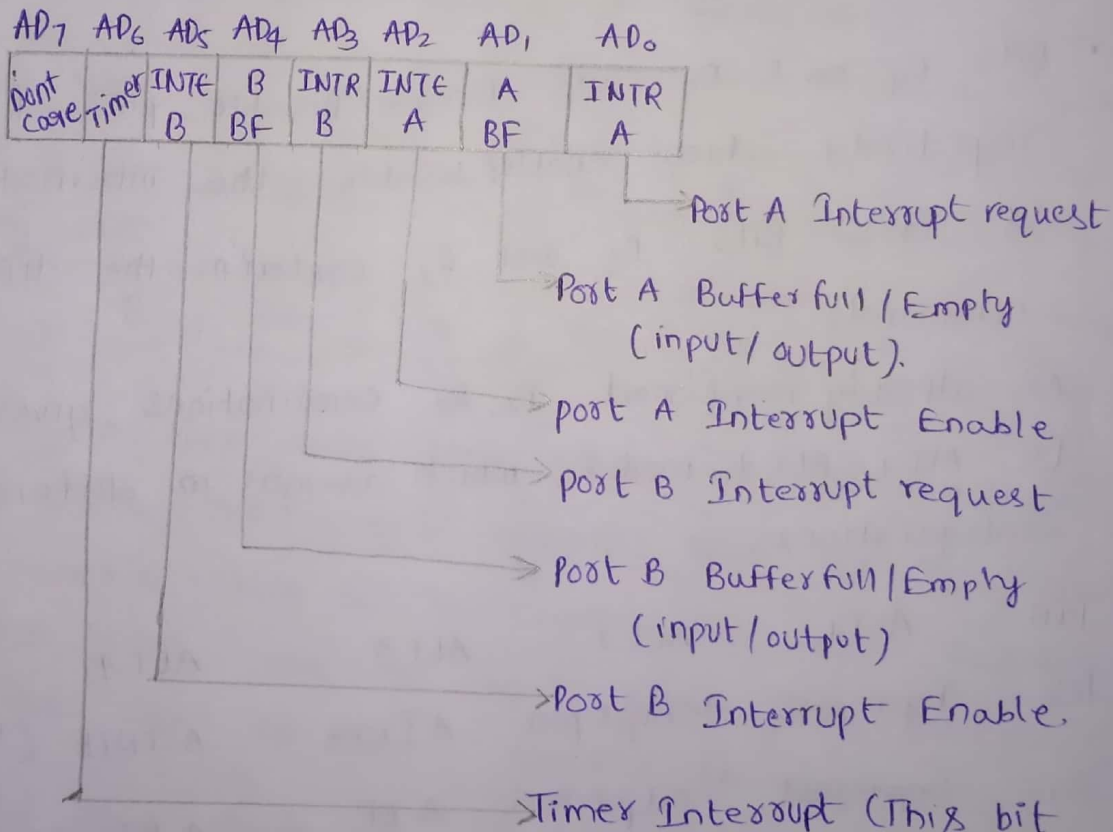
11 = Start - Start timer if it is not running if timer is running - stop at end of count Reload new mode and count and Start again

- The control register contains eight bits
- The content of the lower 8 bits, $D_1 - D_0$ configure Ports A and B as input/output.
- Bits $D_2$ and $D_3$ configure bits $PC_0 - PC_5$ of port C (Port C is a 6-bit while port A and Port B are of (eight) 8-bit) and can have four combinations ALT 1, ALT 2, ALT 3, ALT 4 depending on combinations of $D_2$ and $D_3$.
- Bits $D_4$ and $D_5$ are Enable/Disable pins for A and B respectively which enable/disable the internal flipflop of 8155. Bits $D_6$ and $D_7$ contain the timer commands.

As already mentioned $D_2, D_3$ combinations gives rise to ALT 1 - ALT 4 modes, which assigns in different configurations.

| Pin | ALT 1 | ALT 2 | ALT 3 | ALT 4 |
|---|---|---|---|---|
| $PC_0$ | Input port | Output port | A INTR | A INTR (Port A Interrupt) |
| $PC_1$ | Input port | output Port | A BF | A BF (Port A Buffer Full) |
| $PC_2$ | Input port | output port | A STB | A STB (Port A Strobe) |
| $PC_3$ | Input port | output port | output port | B INTR (Port B Interrupt) |
| $PC_4$ | Input port | output port | output port | B BF (Port B Buffer Full) |
| $PC_5$ | Input port | output port | output port | B STB (Port B Strobe) |

- ALT 1 and ALT 2 correspond to simple input/output of port c respectively.
- In ALT3 mode $PC_0 - PC_2$ bits are used as control signals for port A, while pins $PC_3 - PC_5$ act as output pins.
- In ALT4 mode $PC_0 - PC_2$ bits are used as control signals for Port A, while $PC_3 - PC_5$ bits are used as control signals for Port B.




- It has seven bits. Bit $D_7$ is dont care bit. Bit D6 contain the status of timer. Bits $D_5 - D_3$ concern to status of Port B while bits $D_2 - D_0$ of Port A

## The 8155 timer:

Consists of two 8-bit registers

1. 8-bit LSB and 8-bit MSB
2. In these 16 bits, 14 bits are used for counter and two bits ($M_2$ and $M_1$) for mode selection.
3. The counter is 14-bit down counter. It can operate in four different modes of operation.

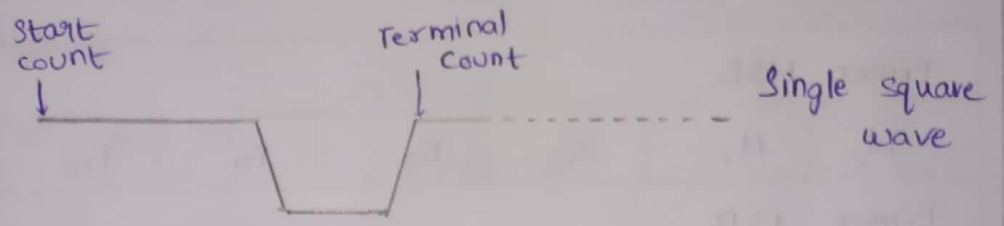
| Timer MSB | | | | | | | |
|---|---|---|---|---|---|---|---|
| $M_2$ | $M_1$ | $T_{13}$ | $T_{12}$ | $T_{11}$ | $T_{10}$ | $T_9$ | $T_8$ |
| Timer LSB | | | | | | | |
| $T_7$ | $T_6$ | $T_5$ | $T_4$ | $T_3$ | $T_2$ | $T_1$ | $T_0$ |

Mode 0: In this mode, timer gives only one cycle of square wave, the output remains high for ½ count and remain slow for ½ count. wave width depends on two factor: one is input clock pulse frequency, and the other is count loaded in counter.

Mode 1: This mode is similar to single square wave in operation but the when counter becomes zero. the count value is automatically reloaded. Thus it provide continuous square wave.

Mode 2: This mode gives a single clock pulse as a output of the end of count. The output is high normally, but it becomes low for 1 clock pulse and again it will become high and remain high.

Mode 4 : This mode is similar to mode 2 but when the counter becomes zero the count value is automatically reloaded. Thus it provide continuous pulse.

| Mode bits $M_2$ | Mode bits $M_1$ | $\overline{\text{Timer out}}$ waveforms |
|---|---|---|
| 0 | 0 |   |
| 0 | 1 |   |
| 1 | 0 |   |
| 1 | 1 |   |

Timer modes and output.

2. Illustrate the interfacing of seven segment LED output ports using 8155.

Port Address

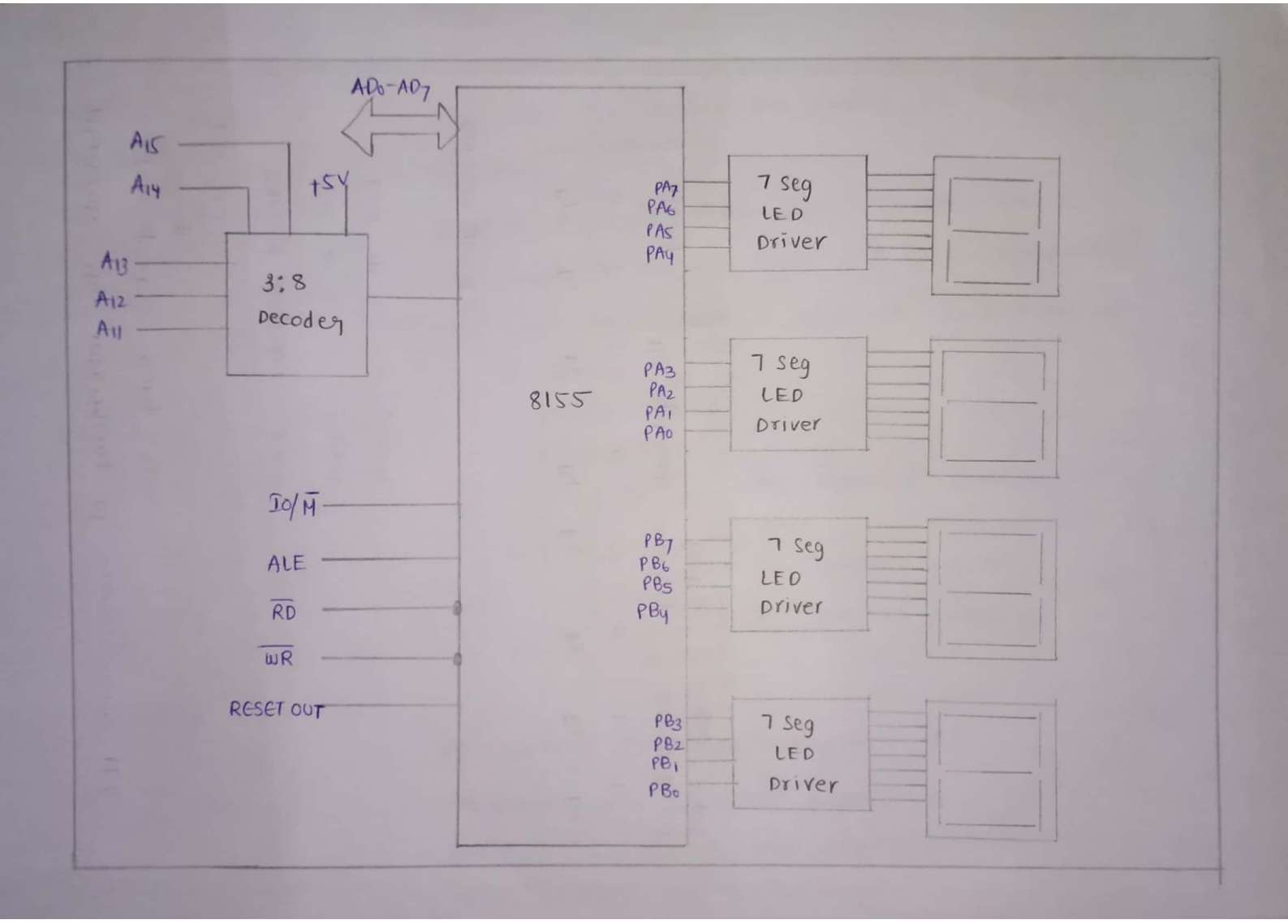
Control Register : 20H

Port A : 21H

Port B : 22H

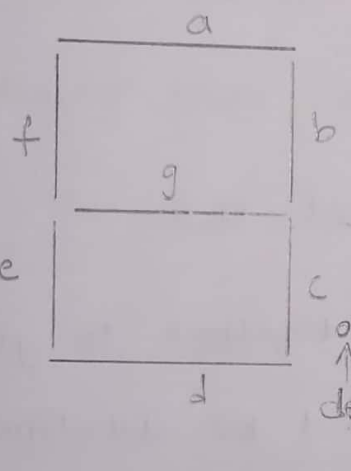
Control word :

| $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| Timer. | | Not applicable | | Use for Port C | | Port B Output | Port A output |



## Seven-segment LED's

- often used to display BCD numbers and a few alphabets.
- A group of eight LEDs physically mounted in the shape of number eight and a decimalpoint.
- Each LED is called a segment and labelled as 'a' through 'g'.

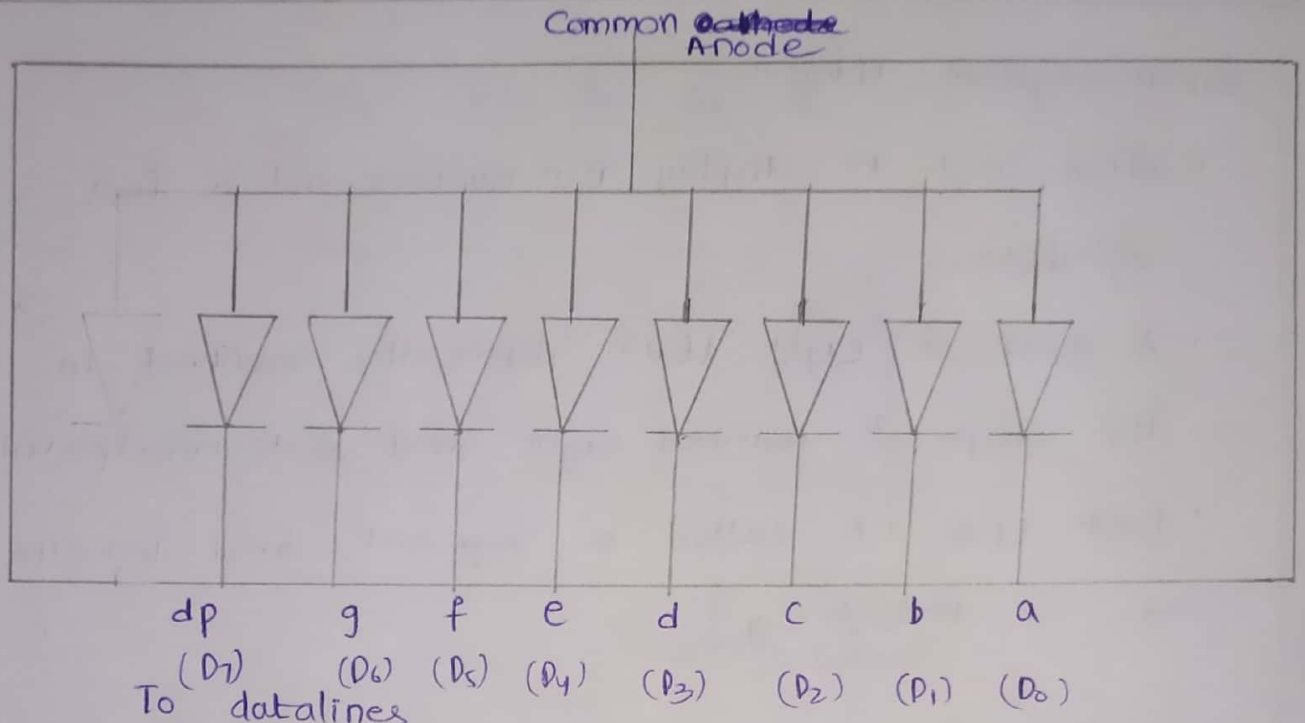



Two types of seven-segment LEDs

1. Common anode
2. Common cathode.

In common anode seven segment LED

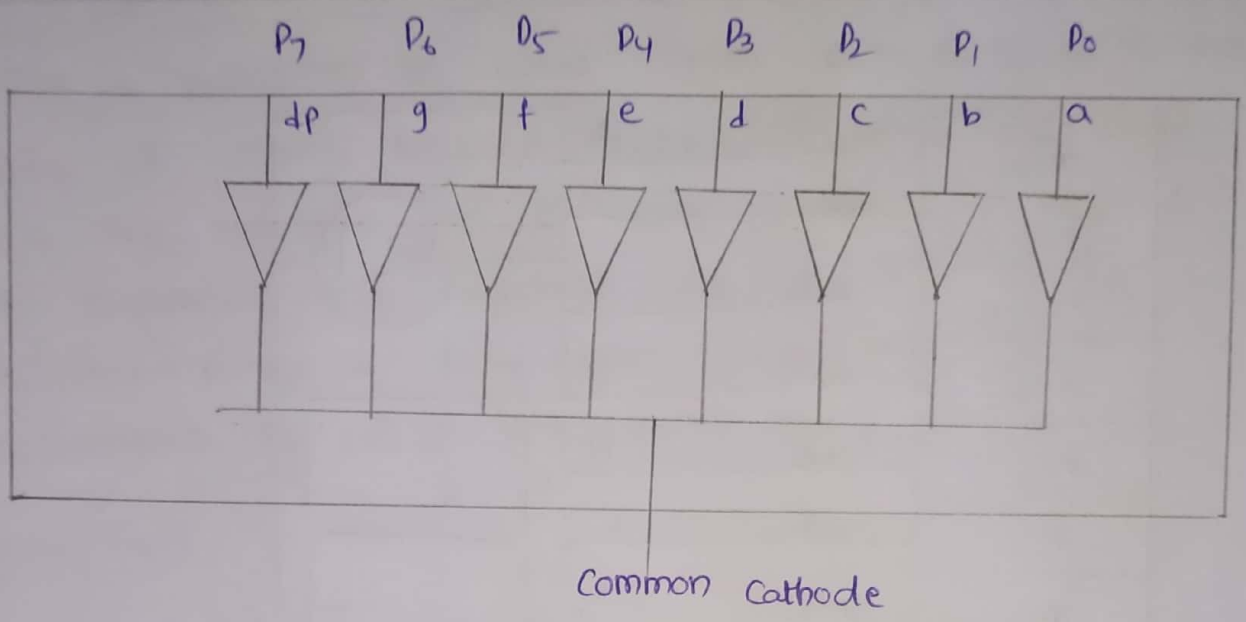
- All anodes are connected together to a power supply and cathodes are connected to datalines.
- logic 0 turns on a segment.
- Example: To display digit 1, all segments except b and c should be off.
- Byte 11111001 = F9H will display digit 1.




To datalines through an interfacing device

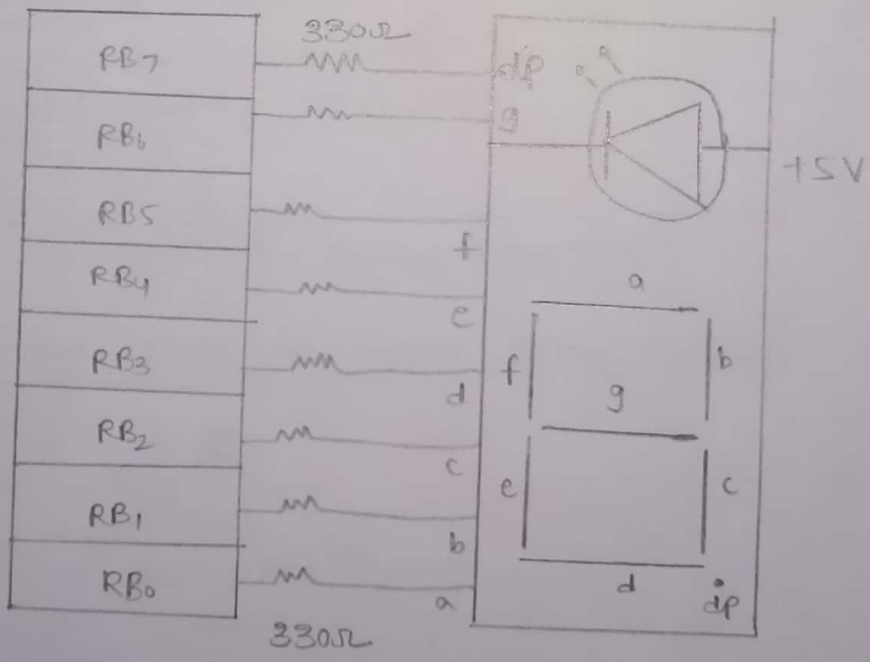
In common cathode seven segment LEDs.

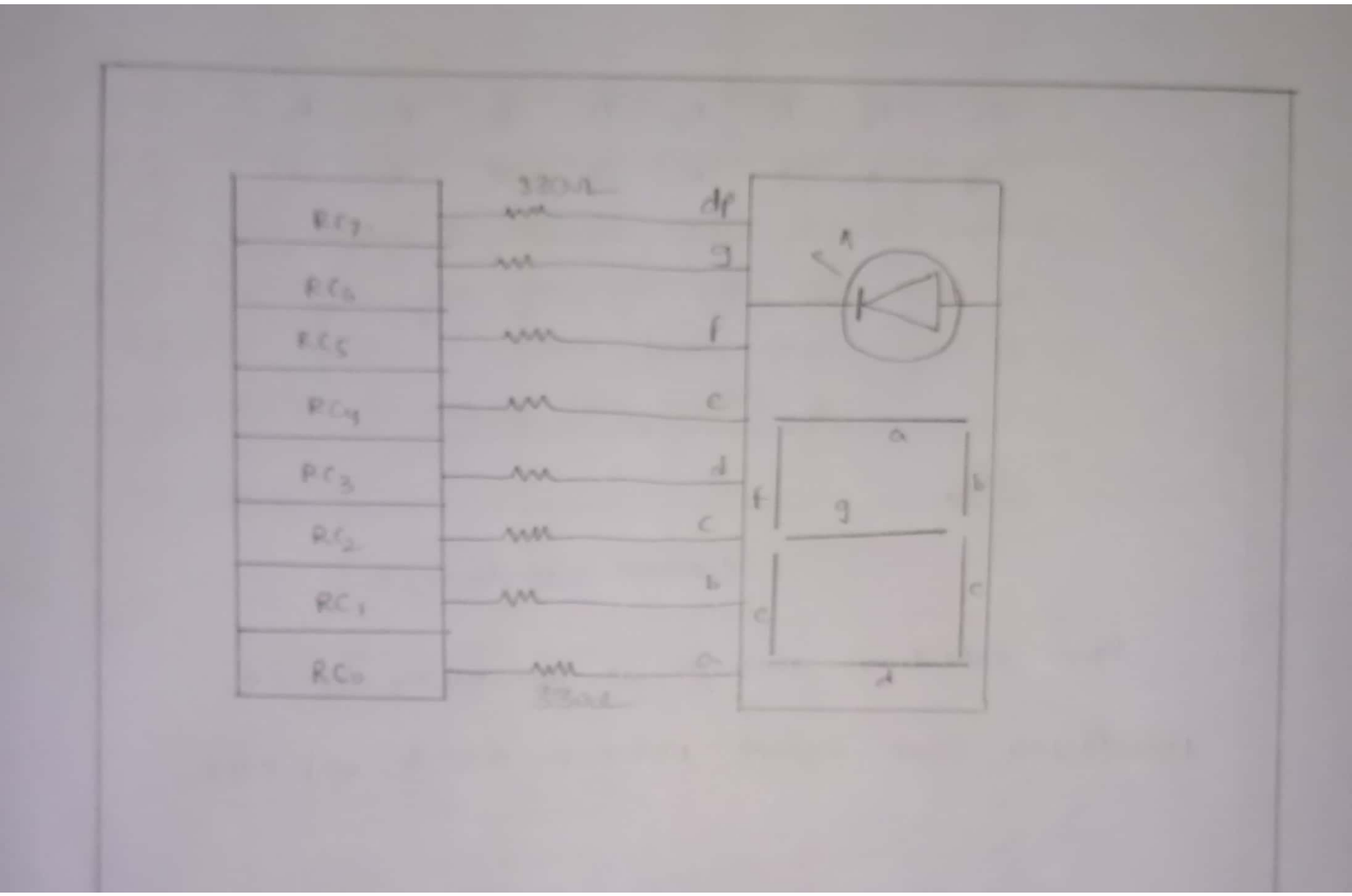
- All cathodes are connected together to ground and the anodes are connected to datalines.
- Logic 1 turns on the segment.
- Example: To display digit 1, all segments except b and c should be off
- Byte 00000110 = 06H will display digit 1.




From data lines through an interfacing device.

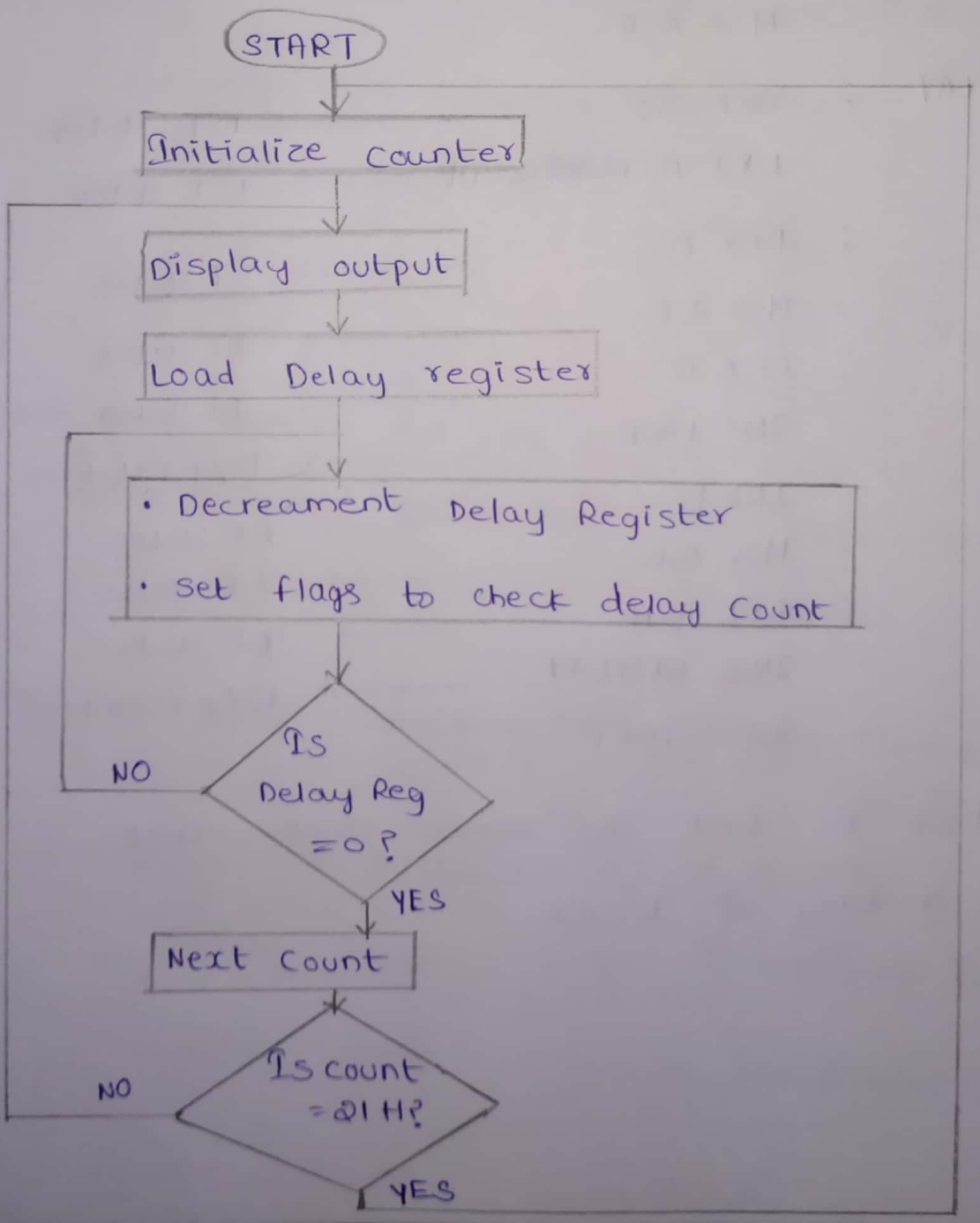
Interfacing Seven-Segment LEDs to PORT B and PORT C





3. write a program to count from 0 to 20 H with the delay of 100ms between each count. After the count 20H, the counter should reset itself and repeat the sequence. Use register pair DE as a delay register. Draw a flow chart and show your calculation to setup the 100ms delay.

Flow chart :



Time delay = 100 ms

A 16-bit register pair (DE) is used as delay

Initial number = Count

Sample program:

```
#ORG   2000H
START  :  MVI B, 00H                          
          MOV A, B                            
DISPLAY :  OUT 00          ------------> 10T states
          LXI D, Count     ------------> 10T states
LOOP   :  DCX D            ------------> 6T states   }
          MOV A, E         ------------> 4T states   }
          ORA D            ------------> 4T states   }
          JNZ LOOP         ------------> 10/7 T states. }
          INR B            ------------> 4T states
          MOV A, B         ------------> 4T states
          CPI 15H          ------------> 7T states
          JNZ DISPLAY      ------------> 10/7 T states.
          JZ START
```

we have to find the number count that cause a delay of 100ms.

## Time delay Calculations

Time delay inside the loop :

No of T states inside loop = (6+4+4+10) T states

⇒ 24 T

This loop runs 'count' no of times

⇒ (24 x Count) T states

Assuming clock frequency = 2 MHz.

$f_{in}$ = 1 MHz

$T = 1/f_{in} \times 10^{-6}$ sec

$T \rightarrow 10^{-6}$ sec

Time delay outside the loop :

No of T states outside loop = (10+10+4+4+7+10) T states

= 45 T states

(100 ms)

total time delay = Time delay inside loop + Time delay outside loop

$100 \times 10^{-3} = 24 \times Count \times 10^{-6} + 45 \times 10^{-6}$

$(Count)_{10} = 4164$ ∴ This is indecimal form.

$(count)_{16} = 1044$ ∴ This is hexadecimal form.

Program :

| Address | Label | Mnemonics | Hexcode | Comments |
|---|---|---|---|---|
| 2000 | START | MVI B, 00 | 06 | // Initialize counter |
| 2001 | | | | |
| 2002 | | MOV A, B | 78 | // Moving to accumulator for output |
| 2003 | DISPLAY | OUT 00 | D3 | // Display at port 00 |
| 2004 | | | 00 | |
| 2005 | | LXI D, 1044H | 11 | // Load DE pair with $(1044)_{16}$. |
| 2006 | | | 45 | for delay |
| 2007 | | | 10 | |
| 2008 | LOOP | DCX D | 1B | // Decreament the content by 1 |
| 2009 | | MOV A, E | 7B | // Move register content of E to A |
| 200A | | ORA D | B2 | // Logical or of Contents A and E to check for zeroes |
| 200B | | JNZ LOOP | C2 | // Jump if not zero. |
| 200C | | | 08 | |
| 200D | | | 20 | |
| 200E | | INR B | 04 | // Increament [B] for next number |
| 200F | | MOV A, B | 78 | // [B] → [A] |
| 2010 | | CPI 15H | FE | // Compare with $(21)_{10}$ |
| 2011 | | | 15 | |

| | | | |
|---|---|---|---|
| 2012 | JNZ DISPLAY | C2 | // Jump to display if $[A] \neq (21)_{10}$. |
| 2013 | | 03 | |
| 2014 | | 20 | |
| 2015 | JZ START | CA | // If $[A] = (21)_{10}$ Again start the |
| 2016 | | 00 | Counting process. |
| 2017 | | 20 | |

NOTE: This is (infinite) continuous process counts from 0 to 20 again resets and continues. So the output is shown at an instant. The process continues as long as we stop execution manually.