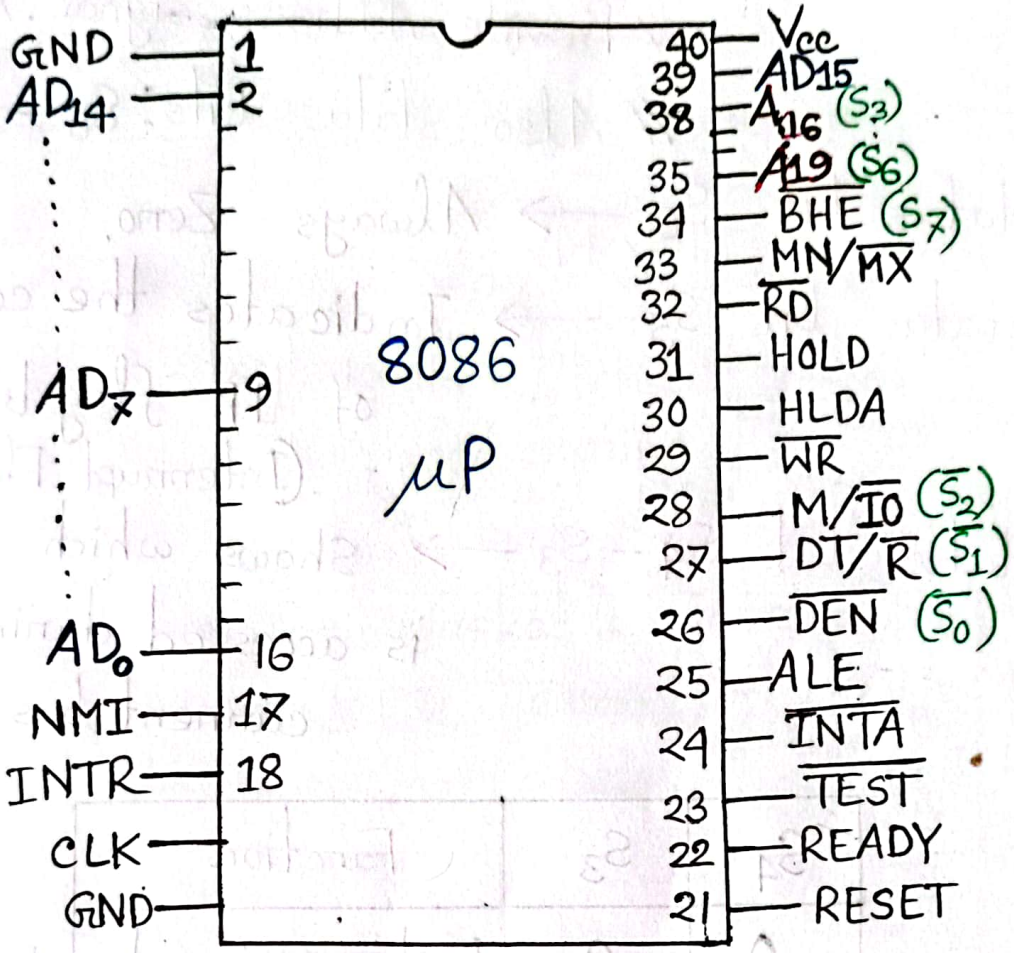
8086 Microprocessor

Pin diagram of 8086 Microprocessor:



Pin Specifications of 8086 Microprocessor:

VCC → Power supply input, provides +5V signal for the microprocess

GND → Ground connection is the return for the Power Supply.

$AD_{15} - AD_0$ → Address/Data bus line compos the multiplexed address or data bu on the 8086.

#Address bus: $A_{15} - A_0$ when ALE = 1.

Data bus: $D_{15} - D_0$ when ALE = 0

Subject:.....
Date:.....Time:.....

A_{19}/S_6 ---- A_{16}/S_3 ⟶ The address/status bus bits are multiplexed.

Provide Address signal: $A_{19}-A_{16}$

Also status bits: S_6-S_3.

Status bit S_6 ⟶ Always Zero.

Status bit S_5 ⟶ Indicates the condition of IF flag bits. (Internupt Flag)

Status bit S_4-S_3 ⟶ Shows which segment is accessed during the current bus cycle.

S_4	S_3	Function	
0	0	Extra Segment	ES
0	1	Stack Segment	SS
1	0	Code Segment	CS
1	1	Data Segment	DS

CLK ⟶ The clock Pin provides timing source for microprocessor clock signal. Must have a duty cycle 33%.

RESET ⟶ Reset input causes the MP

Sweet Bird

to reset itself if the Pin is held high for a minimum four clocking period.

After RESET, instruction execution begins at FFFF0H and the IF Flag is cleared.

MN/$\overline{MX}$ ⟶ Selects the minimum or maximum mode operation.

$\overline{BHE}$/S7 ⟶ Bus high enable pin is used to enable the most significant data bits ($D_{15}-D_8$) during Read or, Write operation. State of S7 is always 1.

NMI ⟶ Non-maskable interrupt input is similar to INTR except that the NMI interrupt does not check if the IF Flag is 1 or 0. If NMI is activated, this interrupt input gives interrupt Vector 2.

$\overline{TEST}$ ⟶ Sends microprocessor in WAIT state. During this time, data comes from co-microprocessor.

When $\overline{TEST}$ = 0 : Microprocessor Waits (WAIT instruction Works)

When $\overline{TEST}$ = 1 : WAIT instruction waits for $\overline{TEST}$ to become a logic 0.

Subject:.....
Date:.....Time:.....

Maximum Mode Pin:

In order to achieve maximum mode for use with external coprocessors;

MN/$\overline{MX}$ must be connected to Ground.

That is: MN/$\overline{MX}$ = 0.

#$\overline{S_2}$, $\overline{S_1}$, $\overline{S_0}$ → Status bits indicate the function of current bus cycle.

$\overline{S_2}$	$\overline{S_1}$	$\overline{S_0}$	Function
0	0	0	Interrupt Acknowledgement
0	0	1	I/O Read
0	1	0	I/O Write
0	1	1	Halt
1	0	0	Opcode Fetch
1	0	1	Memory Read
1	1	0	Memory Write
1	1	1	Passive

Sweet Bird

Subject:.....
Date:.....Time:.....

Minimum Mode Pin:

Minimum mode operation is obtained by connecting MN/$\overline{MX}$ Pin directly to +5V.
It must not be connected through pull-up registen.

$\overline{INTA}$ ⟶ Internupt acknowledgement generated by the MP response to Internupt which causes Internupt Vecton to bepat on the data bus.

ALE ⟶ Address Latch Enable shows that the 8086 address/data bus (A_{15} – A_0) contains address information

$\overline{DEN}$ ⟶ Data Bus Enable activates externnal data bus buffering.

READY ⟶ Ready Pin is related to $\overline{TEST}$.
When TEST is activated, MP is not ready to do work. ⟶ READY = 0
When the function of TEST is finished, MP is ready to do work ⟶ READY = 1.

When $\overline{TEST}$ = 0 ⇒ READY = 0 (MP enten WAIT state)
When $\overline{TEST}$ = 1 ⇒ READY = 1 (MP operates).

Sweet Bird

Subject:.....
Date:.....Time:.....

$\overline{RD}$ ⟶ When $\overline{RD}$ =0, the MP data bus is controlled by the memory on I/O device

$\overline{WR}$ ⟶ When $\overline{WR}$ =0, the MP is outputting data to a Memory on, I/O device.

DT/$\overline{R}$ ⟶ The MP will transmit data when DT/$\overline{R}$ =1 and the MP will receive data when DT/$\overline{R}$=0.

M/$\overline{IO}$ ⟶ Selects Memory on, Input-output mode. This pin shows that-microprocesson address bus contains either a memory address on, I/O pont address.

HOLD ⟶ Hold input requests a direct memory access (DMA).

If HOLD =0 : MP executes software.

If HOLD =1 : MP stops executing software.

HLDA ⟶ Hold acknowledgement shows that Microprocesson has entered the HOLD state.

$\overline{SSO}$ ⟶ This signal is combined with M/$\overline{IO}$ and DT/$\overline{R}$ to decode the function of the current bus cycle.

Sweet Bird

Subject:.....
Date:.....Time:.....

Differences between 8086 and 8088 MP

⇨ A 16 bit data bus on the 8086, an 8-bit data bus on the 8088.

⇨ A $\overline{BHE}$/S7 pin is used in 8086, an $\overline{SSO}$ pin is used on the 8088 MP instead of it.

⇨ 8086 MP has M/$\overline{IO}$ pin whereas 8088 MP has IO/$\overline{M}$ pin.

Why $AD_{15} - AD_0$ Pins are Multiplexed?

(i.) To reduce the Pin number.

(ii.) To reduce the cost.

(iii.) To reduce the chip size.

(iv.) To reduce the loading effect (To reduce propagation delay).

Why Data buses are De-multiplexed or, why not leave the buses multiplexed?

→ Memory and I/O require that the address remain valid and stable throughout a read or write cycle.

→ If the buses are multiplexed, the change in address occurs at the memory and I/O, which causes them to read or write data in the wrong locations.

Sweet Bird

Subject:.....
Date:.....Time:.....

Bus Cycle/Machine Cycle:

- It is the time required for one bus operation.
- The time is required for doing any Particular task using bus is called bus cycle.

1 bus cycle = 4 Clock Period = 4T

= (4 × 200) ms = 800 ms.

$\therefore f = \frac{1}{T} = 5\,MHz.$ Clock frequency $= \frac{1}{200 \times 10^{-3}} = 5\,MHz$

Problem:

For the following instruction, find out the value of control pin. MOV AX, BX

(ALE, $\overline{DEN}$, $\overline{RD}$, $\overline{WR}$, M/$\overline{IO}$, DT/$\overline{R}$)

Solution: This is a READ operation from memory.

ALE = 0

$\overline{DEN}$ = 0 – because data comes from memory to register

$\overline{RD}$ = 0 – because read operation

$\overline{WR}$ = 1

M/$\overline{IO}$ = 1 – because memory operation

DT/$\overline{R}$ = 0

During Read operation, $\overline{DEN}$ enables external devices to supply data to the µP.

Sweet Bird

Subject:.....
Date:.....Time:.....

Problem For the following instruction, find out the value of control pins: Memory Map Interfacing

I/O Interfacing ← OUT DX, AX. or; MOV [DX], AX

Solution: This is a WRITE operation to the external I/O device. So-

$ALE = 0$ ⟶ For both READ and WRITE operation, $ALE = 0$, $\overline{DEN} = 0$

$\overline{DEN} = 0$

$M/\overline{IO} = 0$ || For MOV operation, $M/\overline{IO} = 1$.

$DT/\overline{R} = 1$

$\overline{WR} = 0$; $\overline{RD} = 1$.

Problem Find out the value of control pins for-

MOV [BX], AX.

Solution: This is also WRITE operation, but to the memory. Thus-

$ALE = 0$

$\overline{DEN} = 0$

$\overline{RD} = 1$

$\overline{WR} = 0$

$M/\overline{IO} = 1$

$DT/\overline{R} = 1$.

Sweet Bird

Subject:
Date: Time:

Problem Find out the values of control pin for-

IN AX, DX.

Solution: This is a READ operation from I/O device.

So-

$ALE = 0$

$\overline{DEN} = 0$

$DT/\overline{R} = 0$

$M/\overline{IO} = 0$

$\overline{RD} = 0$

$\overline{WR} = 1$.

What will be the values of control pins for MOV CX, DX? [Data Movement Instruction]

⇨ In this case, the controls pins are unidentified because it is an internal register to register movement. (Neither Read nor Write operation)

What will be value of control pins after PUSH AX?

⇨ In this case, contents of AX copies to stack segment that is an internal register operation. So in this case, control pins cannot be identified.

Sweet Bird

Subject:.....
Date:.....Time:.....

T State

One period of clock frequency. Usually, $T = 200\,ns$ for 8086 MP.

Instruction Cycle:

The time microprocessor requires to fetch and execute an instruction is referred to as an instruction cycle.

An instruction cycle consists of one or more machine cycle/bus cycle.

Bus Buffering and Latching:

→ Computer systems have three buses:

- Address
- Data
- Control

→ The ALE pin controls a set of latches.

→ Abl. signals must be buffered.

Latches buffer for $A_0 - A_{15}$.

→ Control and $A_{16} - A_{19} + \overline{BHE}$ are buffered separatedly.

→ Data bus buffering must be bi-directional.

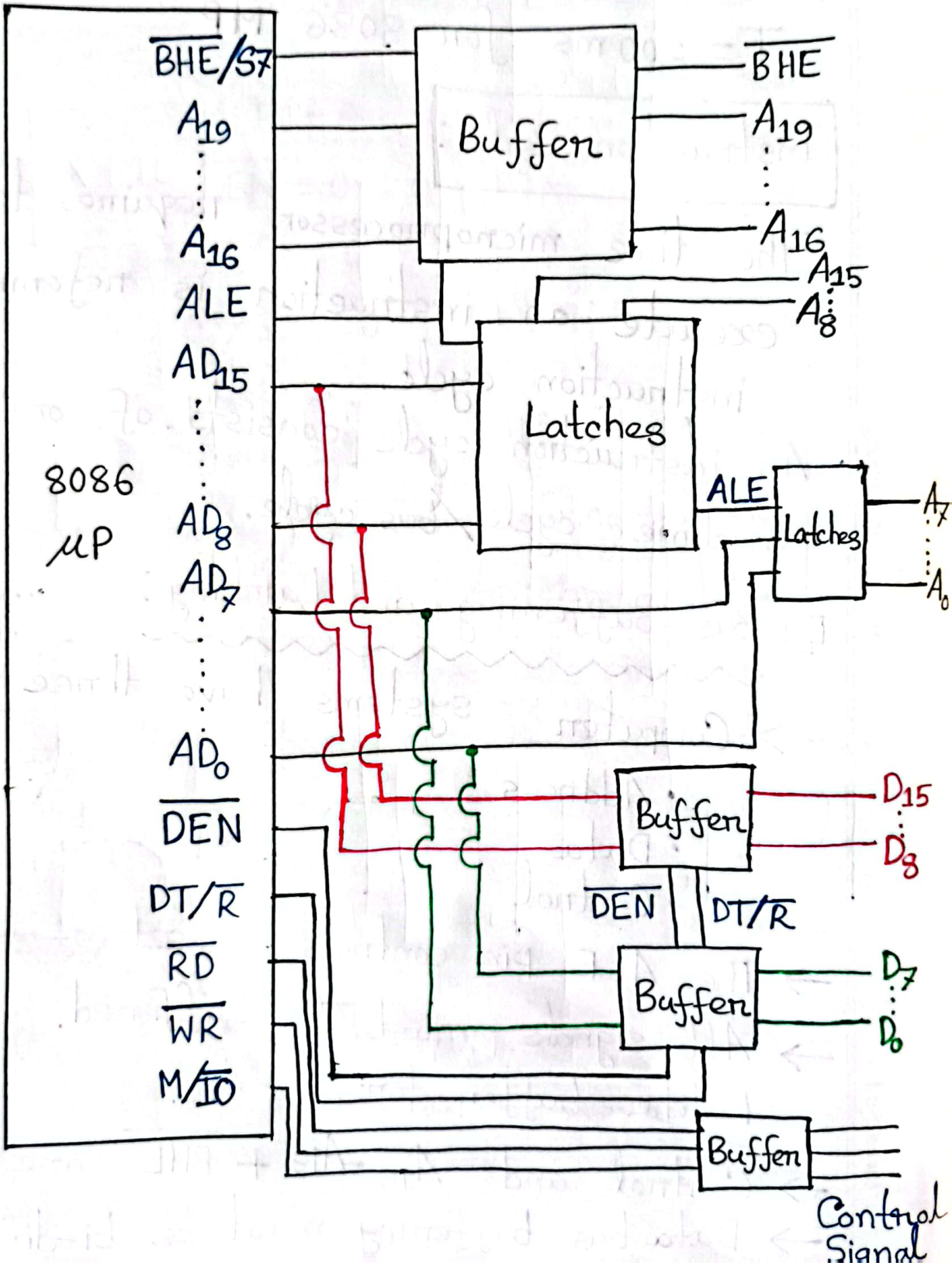
Subject:.....

Date:.....Time:.....

Bus Buffering and Latching Structure in 8086

→ Latches buffer to $A_{15} - A_0$.

→ $\overline{BHE}$ selects the higher order Memory Bank.



Sweet Bird

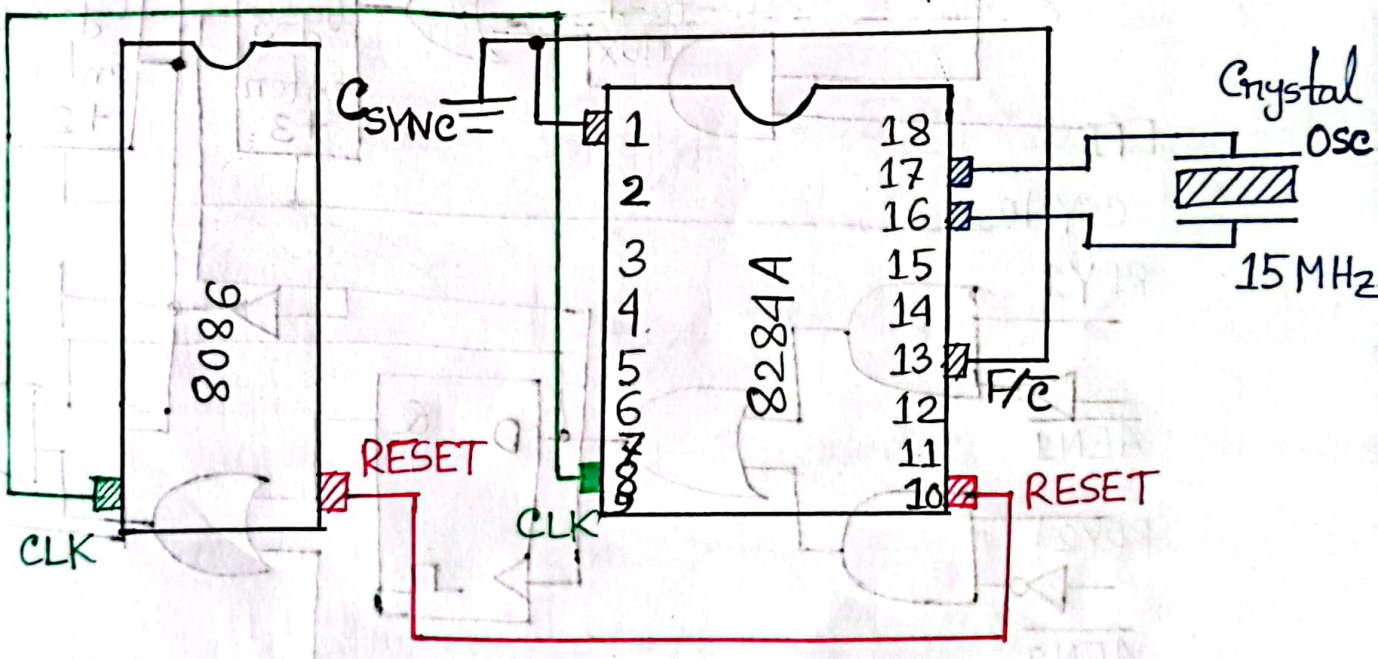
Subject:.....
Date:.....Time:.....

8284A Clock Generaton

Basic function of Clock generaton:

1.) Clock generation.
2.) RESET Synchronization.
3.) READY synchronization.
4.) Peripheral clock signal.

Connection of the 8284 and the 8086:



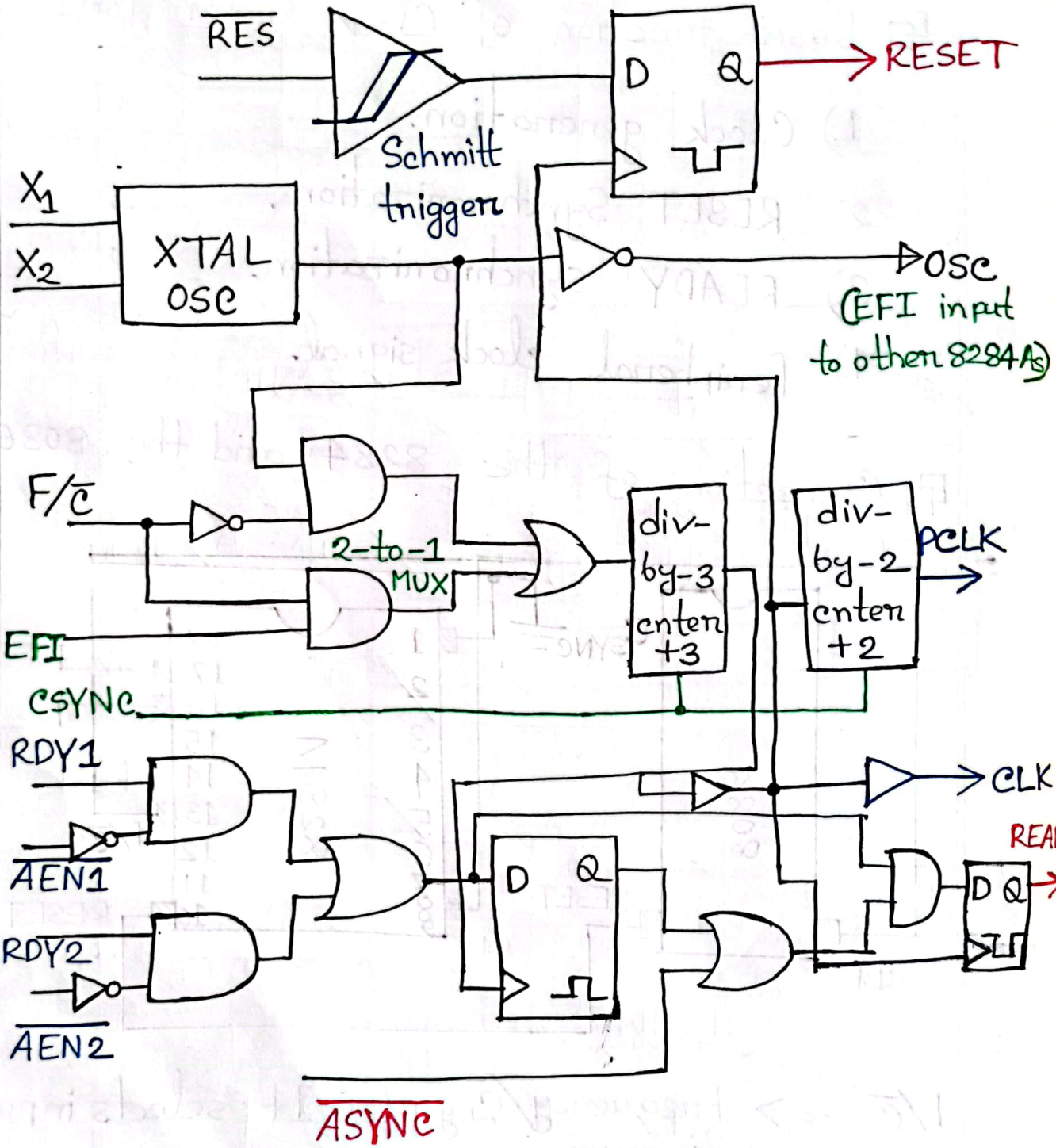
F/C → Frequency/Crystal. It selects input chooses for the clocking source for the 8284A.

Sweet Bird

Subject:.....

Date:.....Time:.....

8284A Clock Generaton Circuit



Sweet Bird

Subject:.....
Date:.....Time:.....

Clock generation Process:

→ Crystal is connected to X_1 and X_2.

→ XTAL OSC generates a square wave signal at crystal's frequency which feeds:

- An inverting buffer (output OSC) which is used to drive the EFI input of other 828As.
- 2-to-1 MUX

  F/$\overline{C}$ selects, XTAL on EFI external input.

→ The MUX drives a divide-by-3 counter which converts 15 MHz to 5 MHz clock signal.

→ This drives:

- The READY flipflop (READY Synchronization).
- A second divide-by-2-counter which generates a 2.5 MHz clk for peripheral components. (PCLK)
- The RESET flop.
- CLK which drives the 8086 CLK input.

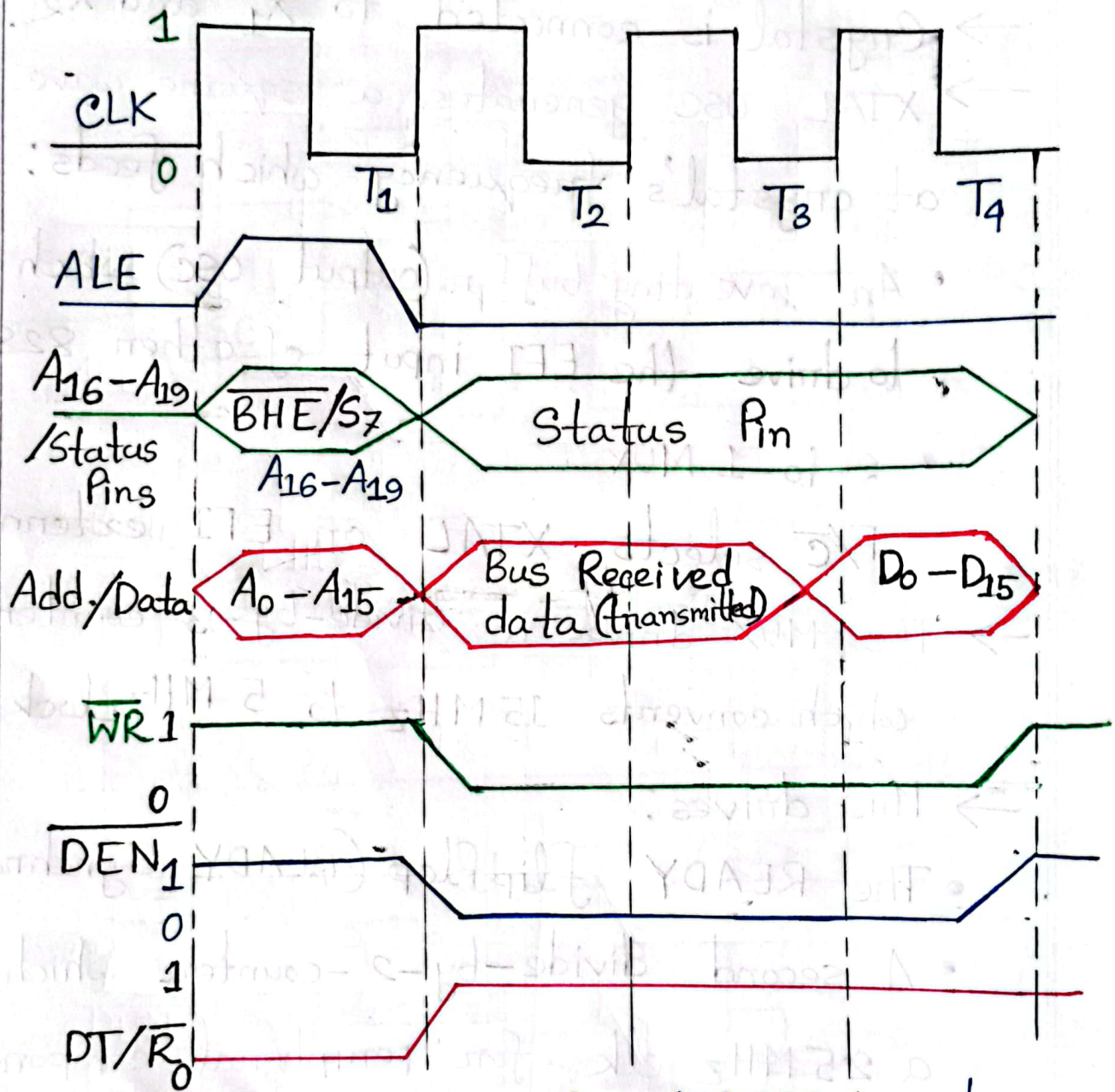
→ Here, when $\overline{ASYNC}$ =0: Two-stage synchronization.

when $\overline{ASYNC}$ =1: One-stage synchronization.

Sweet Bird

Subject:.....
Date:.....Time:.....

Writing BUS Cycle:



Timing Diagram for WRITE Operation

Instructions
- OUT DX, AX → I/O
- MOV [DX], AX → Memory

Sweet Bird

Subject:.....
Date:.....Time:.....

During T_1:

→ During the first clock cycle T_1, the address of the memory on, I/O location is sent through the address bus and address/idata bus connection.

→ Control signals ALE, $M/\overline{IO}$ and $DT/\overline{R}$ latch the address on the address bus and set the direction of data transfer on data bus.

During T_2:

→ 8086 issues the WRITE signal.

→ Data will be written to on received by data bus.

→ Memory on, input output device begin to perform WRITE.

During T_3:

→ Clocking period is provided to allow the memory time to access data.

During T_4:

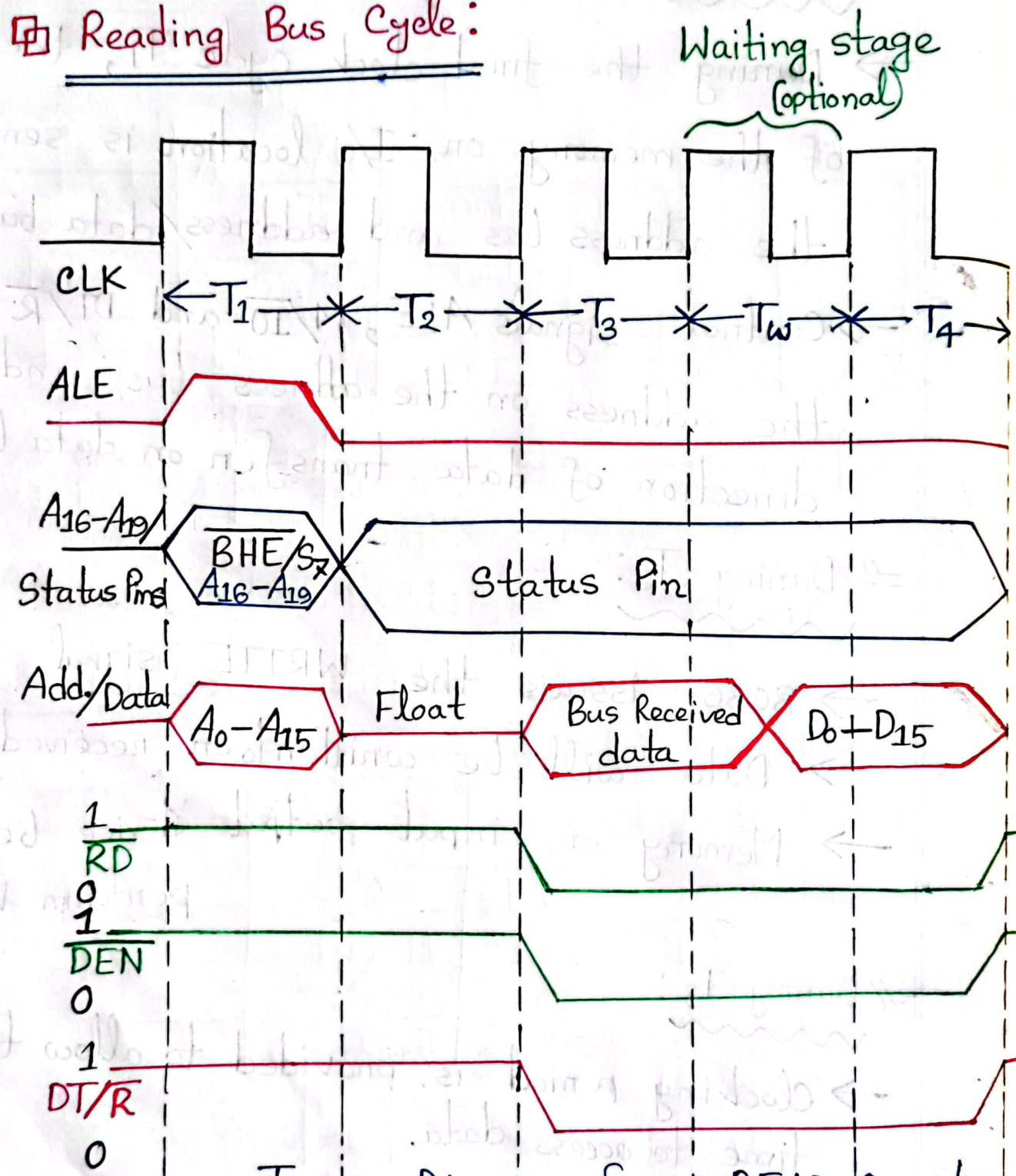
→ D_0-D_{15} data buses WRITE the data to the memory on I/O port address.

→ All the buses are deactivated for the preparation

Sweet Bird

of next bus cycle.

Reading Bus Cycle:



Timing Diagram for READ Operation

For Instructions → IN AX, DX → I/O
→ MOV AX, [DX] → Memory

Sweet Bird

Subject:.....
Date:.....Time:.....

During T_1:

→ During the first clock period, the address of the memory or input-output location is sent through the address bus and address/data bus.

→ Control signal ALE, $M/\overline{IO}$, $DT/\overline{R}$ latch the addresses on the address bus and set the data transfer direction on data bus.

During T_2:

→ Microprocessor takes time to setup the data to be read.

→ This stage is defined as float stage.

During T_3:

→ 8086 issues the read signal.

→ Microprocessor accepts the data to be read from memory or, I/O device.

During T_4:

→ All the signals are deactivated for the preparation of next bus cycle.

Sweet Bird

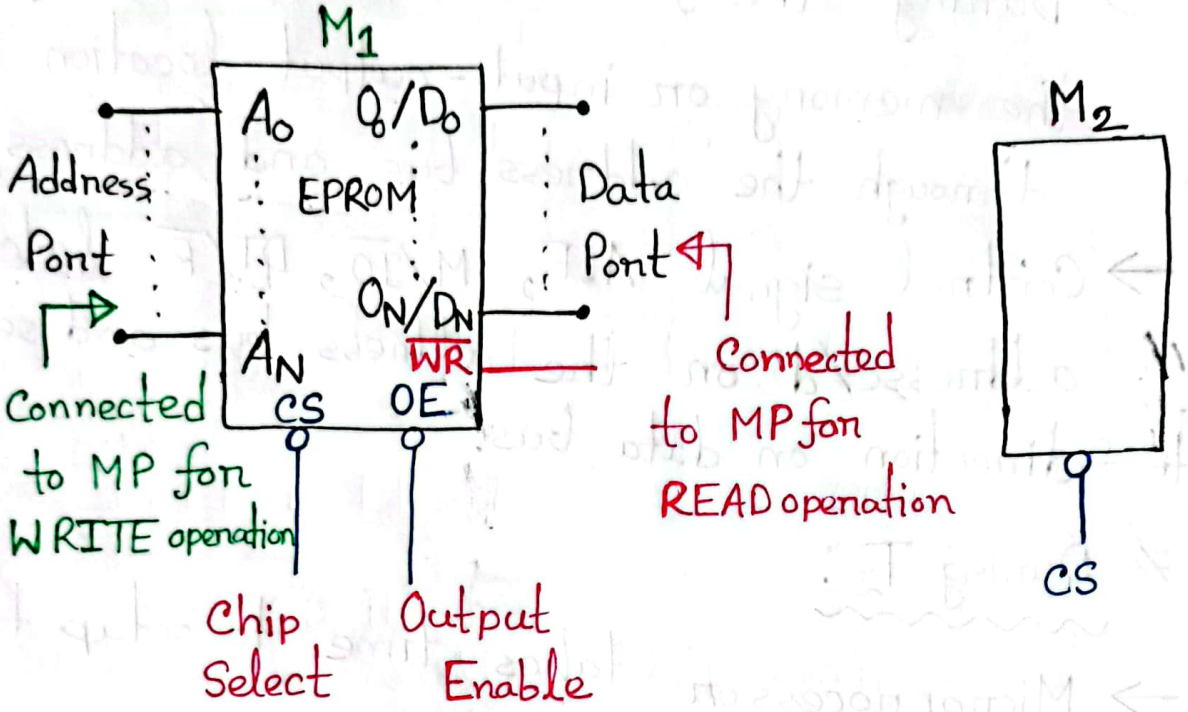
Memory Map Interfacing

Subject:.....
Date:.....Time:.....

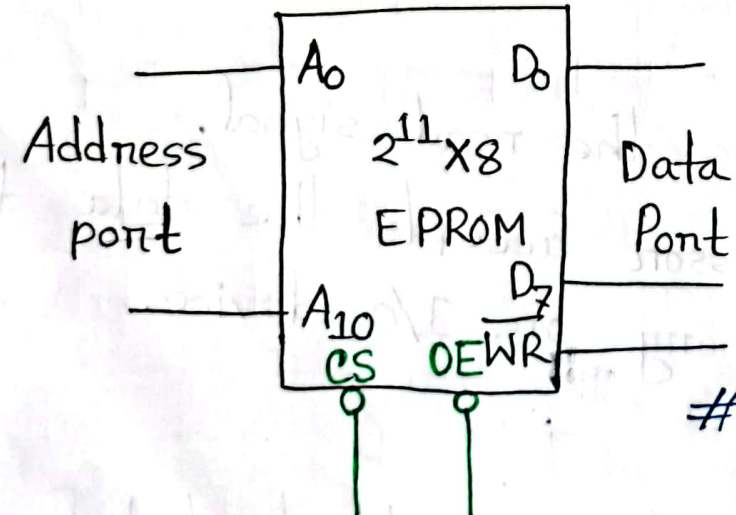
#Chapter:03

[Ramesh Gaonkar]

#Simple Block diagram for memory unit:



Example: A 2^{11} x8 EPROM:

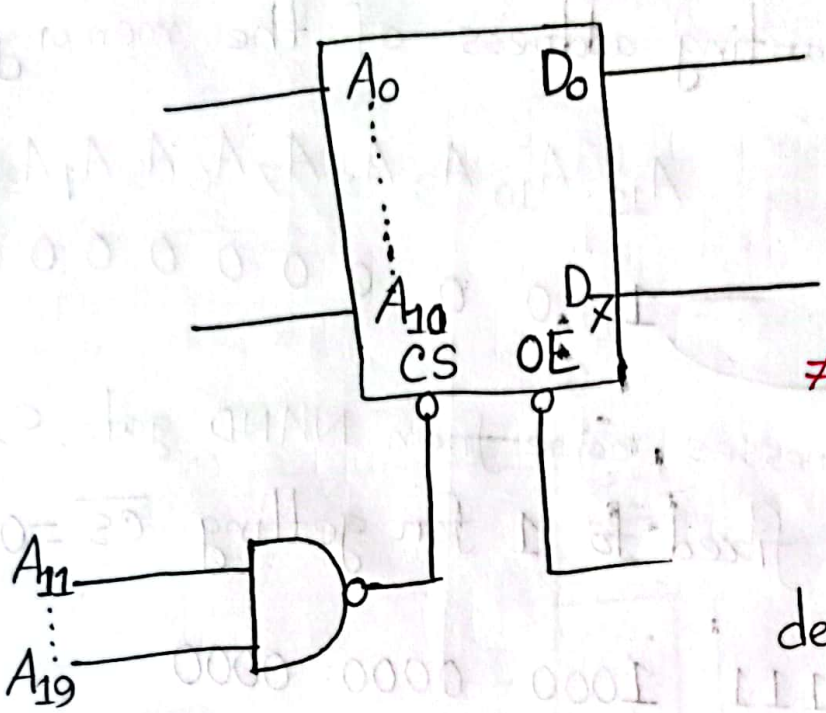


#We know that- 8086 Microprocessor has 20 bit address bus.

#But the 2^{11} x8 EPROM (Memory Device) has 11 bit address bus.

So, the rest of the 9-bit address line is provided through a 3-to-8 line decoder or a NAND gate.

Sweet Bird



To generate more than one address by A_{11} ..... A_{19}, decoder is used.

Problem For the given memory device–

(i.) Find the size of memory

(ii.) Starting address of memory

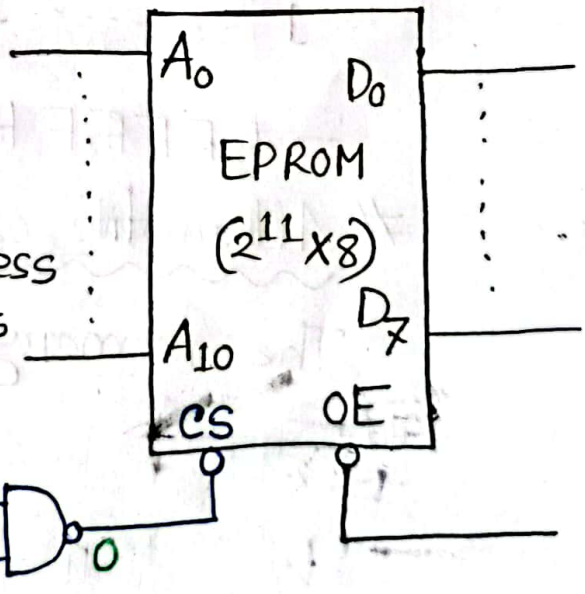
(iii.) Last address of memory.

(i.) The size of a memory device $= 2^n$

where n = Number of Address ports

Here, $n = 11$.

∴ The size of the memory device $= 2^{11} = 2048$.



Subject:.....
Date:.....Time:.....

(ii) The starting address of the memory

$A_{19}\ A_{18} \ldots\ldots A_{11}\ A_{10}\ A_9\ A_8\ A_7\ A_6\ A_5\ A_4\ A_3\ A_2\ A_1\ A_0$

1 1 ..... 1 0 0 0 0 0 0 0 0 0 0

These addresses come from NAND gate. So, they are fixed to 1 for getting $\overline{CS} = 0$.

= 1111 1111 1000 0000 0000

= FF800 H.

Starting address = FF800 H.

(iii) The last address of the memory

$A_{19} \ldots\ldots A_{11}\ A_{10}\ A_9\ A_8\ A_7\ A_6\ A_5\ A_4\ A_3\ A_2\ A_1\ A_0$

1 ..... 1 1 1 1 1 1 1 1 1 1 1

= FFFFF H.

Alternative way:

The memory size $= 2^{11} = 2048.$

$= 0800H$

$= 0000H + (07FFH)$

Displacement or Range

Sweet Bird

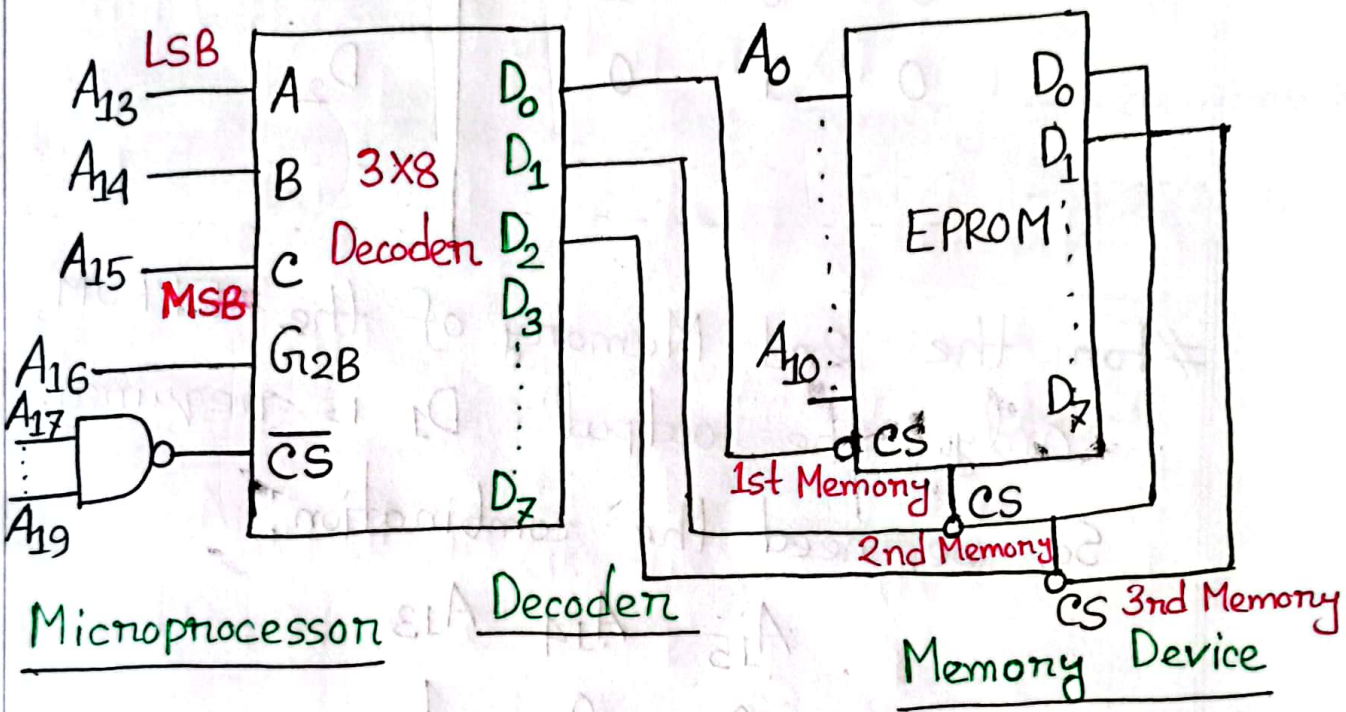
Subject:.....
Date:.....Time:.....

(Starting Address + Displacement)

∴ Last address of memory = FF800H + 07FFH

= FFFFFH.

Problem:



Find the initial and the final address of the 2nd memory.

Here, A_{11} and A_{12} are don't care since they are not present.

Size of the 2nd memory $= 2^{11} = 2048 = 0800H$

∴ 2nd Memory Size = 0000H + 07FFH.

Sweet Bird

Subject:.....
Date:.....Time:.....

The input port A, B and C of the decoder functions as -

C A_{15}	B A_{14}	A A_{13}	O/P
0	0	0	D_0
0	0	1	D_1
0	1	0	D_2
.....	.....	.....	.....
.....	.....	.....	.....

#For the 2nd Memory of the EPROM, only the output D_1 is required.

So, we need the combination:

A_{15}	A_{14}	A_{13}
0	0	1

∴ Initial Address of the 2nd Memory

$A_{19}\ A_{18}\ A_{17}\ A_{16}\ A_{15}\ A_{14}\ A_{13}\ A_{12}\ A_{11}\ A_{10} \cdots\cdots A_0$

= 1 1 1 1 0 0 1 0 0 0 0

(1 1 1 1: Always = 1; 0 0: Assume Don't care = 0)

= 1111 0010 0000 0000 0000

= F2000 H.

Sweet Bird

Subject:.....
Date:.....Time:.....

∴ Final address of the 2nd Memory

$A_{19}\ A_{18}\ A_{17}\ A_{16}\ A_{15}\ A_{14}\ A_{13}\ A_{12}\ A_{11}\ A_{10}\ A_9 \ldots\ldots A_0$

$= 1\ \ 1\ \ 1\ \ 1\ \ 0\ \ 0\ \ 1\ \ 0\ \ 0\ \ 1\ \ 1 \ldots\ldots 1$

$= 1111\ \ 0010\ \ 0111\ \ 1111\ \ 1111$

$= F27FFH.$

∴ Final address of 2nd Memory = F2000H + 07FFH

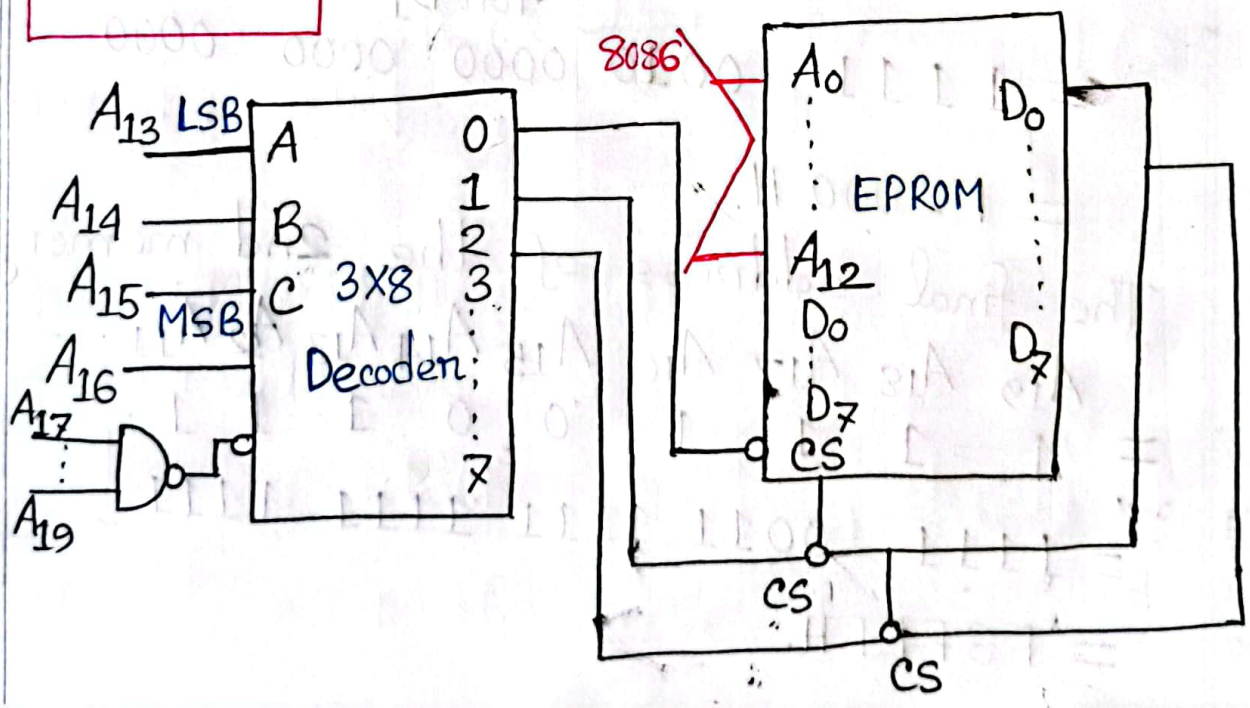
= F27FFH.

#Note that:

Total memory size of the EPROM

$= 3 \times 2^{11}$.

Problem :



Sweet Bird

Subject:.....
Date:.....Time:.....

Question - 01: Find the initial and the final address of 2nd Memory.

Question - 02: Write an instruction to read 2-Byte data from EPROM and write into DX register (from the initial address of 2nd memory 2000H)

Question - 03: What is the size of the memory?

(1.) The initial memory address of the 2nd memory

$A_{19}\ A_{18}\ A_{17}\ A_{16}\ A_{15}\ A_{14}\ A_{13}\ A_{12}\ A_{11} \ldots\ldots\ldots A_0$

$= 1\ \ 1\ \ 1\ \ 1\ \ \underbrace{0\ \ 0\ \ 1}_{\text{For } D_1}\ \ 0\ \ 0 \ldots\ldots\ldots 0$

$= 1111\ \ 0010\ 0000\ \ 0000\ \ 0000$

$= F2000\ H.$

The final address of the 2nd memory

$A_{19}\ A_{18}\ A_{17}\ A_{16}\ A_{15}\ A_{14}\ A_{13}\ A_{12}\ A_{11} \ldots\ldots\ldots A_0$

$= 1\ \ 1\ \ 1\ \ 1\ \ 0\ \ 0\ \ 1\ \ 1\ \ 1 \ldots\ldots\ldots 1$

$= 1111.\ 0011\ 1111\ 1111\ 1111$

$= F3FFFH.$

Sweet Bird

② For 2nd memory,

initial physical address = F2000 H

= F000 X 10H + 2000 H.

Segment address = F000 H. → DS

Offset address = 2000 H. → BX

The program:

```
MOV AX, F000 H
MOV DS, AX
MOV BX, 2000H
MOV DX, [BX]
```

Or:
```
MOV SI, 2000H
MOV DX, [SI]
```

↑ 2 Byte data transfered to DX

#For 1-byte data transfer:

```
MOV DL, [BX]
```

or, `MOV DH, [BX]`

③ Size of 2nd memory $= 2^{13}$.

Total memory size $= 3 \times 2^{13}$.

Subject:.....
Date:.....Time:.....

Bus Timing:

→ Each bus cycle on the 8086 equals four system clock periods. (T states)

→ Since the clock rate is 5 MHz, the transfer rate is = 1·25 MHz.

Difference between Maskable and Non-Maskable Interrupt

Maskable Interrupts	Nonmaskable Interrupts
1.) Can be masked off or made pending.	1.) Cannot be masked off or made pending.
2.) They cannot disable any nonmaskable interrupt.	2.) They can disable any maskable interrupts.
3.) Lower priority than nonmaskable interrupts.	3.) Higher priority than maskable interrupt.
4.) May be vectored or non-vectored.	4.) All are vectored.
5.) Response time is high.	5.) Response time is low.

Sweet Bird

Subject:.....
Date:.....Time:.....

Differences between Minimum and Maximum Mode?

Maximum Mode	Minimum Mode
1.) Multiple processor mode. Along with 8086, other processors like 8087, 8089 etc. are used.	1.) It is a uni processor mode 8086 is the only processor in the system.
2.) Separate bus controller (8288) is used in the maximum mode.	2.) In this mode, no separate bus controller is required.
3.) ALE, $\overline{DEN}$, DT/$\overline{R}$, $\overline{INTA}$ are not directly available and are generated by bus controller 8288.	3.) ALE, $\overline{DEN}$, DT/$\overline{R}$ and $\overline{INTA}$ signals are directly available
4.) The speed of processing is higher.	4.) The processing speed is lower since more load on one microprocessor.
5.) The circuit is more complex but supports multiprocessing.	5.) The circuit is simpler but does not support multiprocessing.

Sweet Bird

Subject:.....
Date:.....Time:.....

Differentiate between Buffer and Latch:

Buffer	Latch
i.) Buffer is bidirectional or unidirectional.	i.) Latch is unidirectional.
ii.) Used with data.	ii.) Used with address.
iii.) Sends data quickly to the microprocessor.	iii.) Latch keeps the stack until next clock comes.

Problem: From the following figure determine

(i.) Initial and final address of the 2nd section.

(ii.) Size of the Memory device.

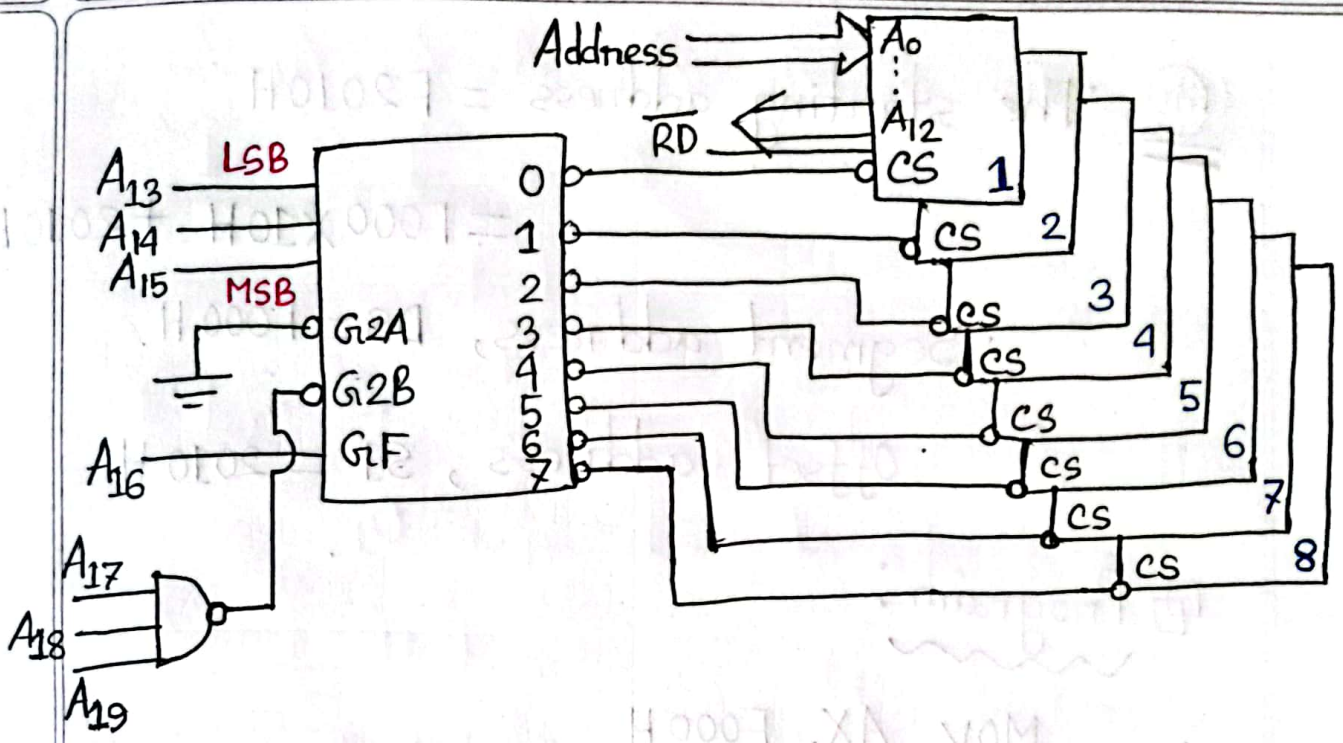
(iii.) Write an instruction to load a final byte in DL and next 2 byte in DX.
The starting address = F2010 H.

(iv.) For initial address from the circuit, record 1 byte data.

Sweet Bird

Subject:.....

Date:.....Time:.....



Solution: (i) The initial address of 2nd section

	A_{19}	A_{18}	A_{17}	A_{16}	A_{15} (MSB)	A_{14}	A_{13} (LSB)	A_{12}	A_{11}	.....	A_0
	↓	↓	↓	↓	↓	↓	↓	↓	↓		↓
=	1	1	1	1	0	0	1	0	0	.....	0

= F2000 H.

The final address of 2nd Section

	A_{19}	A_{18}	A_{17}	A_{16}	A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	.....	A_0
	↓	↓	↓	↓	↓	↓	↓	↓	↓		↓
=	1	1	1	1	0	0	1	1	1	.....	1

= F3FFF H.

(ii) Since there are 13 address ports from A_0 to A_{12}.

∴ Size of the memory $= 2^{13}$.

Sweet Bird

(iii) The starting address = F2010H

= F000 X 10H + 2010H

∴ Segment address, DS = F000 H.

Offset address, SI = 2010 H.

Program:

```
MOV AX, F000H
MOV DS, AX
MOV SI, 2010H
MOV DL, [SI]
INC SI
MOV DX, [SI]
INC SI
INC SI
```

(iv) The initial address = F2000 H

= F000H X 10 + 2000 H.

Segment address, DS = F000 H.

Offset address, BX = 2000 H.

Subject:.....
Date:.....Time:.....

Program:

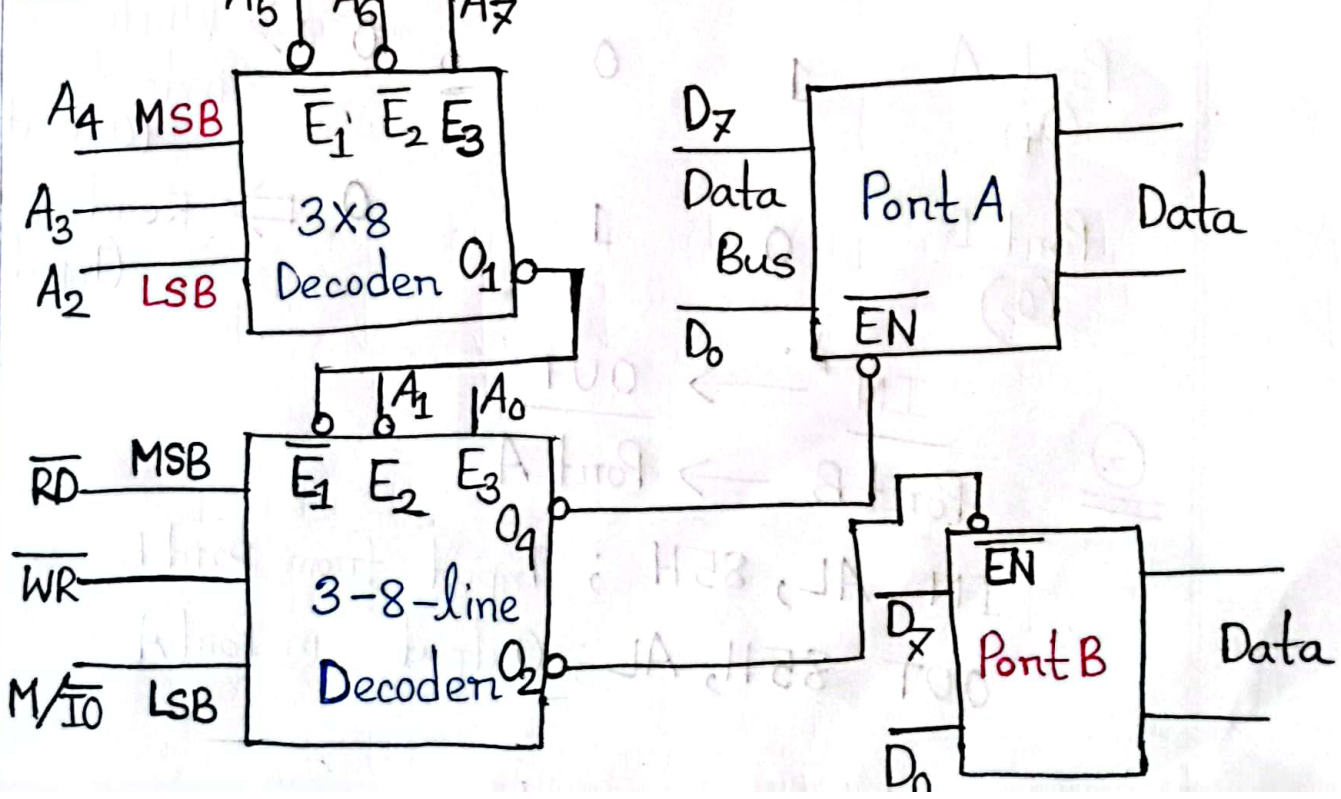
```
MOV  AX, F000H
MOV  DS, AX
MOV  BX, 2000H
MOV  DL, [BX]
INC  BX
```

Problem:

(i) What are the addresses of port A and portB?

(ii) Identify Port A and Port B as input or output Port

(iii) Write program regarding the decoder circuit.



Sweet Bird

Subject:.....
Date:.....Time:.....

① A_7 A_6 A_5 A_4 A_3 A_2 A_1 A_0

1 0 0 0 0 1 0 1 = 85H.

Selects O_1

Here, outputs O_4 and O_2 of 2nd 3X8 decoder depends upon O_1.

∴ Port addresses for both Port A and Port B is = 85H.

Here, A_{19} ..... A_8 = 0 (don't care).

②

	$\overline{RD}$	$\overline{WR}$	$M/\overline{IO}$	
Port A (O_4)	1	0	0	⇒ Write Operation into the I/O port. (Output Port)
Port B (O_2)	0	1	0	⇒ Read operation (Input Port)

IN ⟶ OUT

③ Port B ⟶ Port A

```
IN  AL, 85H ; Input from port B
OUT 85H, AL ; Output in Port A.
```

Sweet Bird

8255 PPI

Programmable Peripheral Interface

Subject:.....

Date:.....Time:.....

Parallel I/O device:

- Displays
- Keyboard
- Printers (Old Printer) etc.

Serial I/O devices:

- Some printers
- Data communication.

8255 PPI

→ 24 I/O pins.

→ 8 bit parallel ports : A and B.

→ C Port: can be grouped as 4 bit CU (PC_4-PC_7) on C Upper and CL (PC_3-PC_0) on, C Lower.

Basically two modes :

BSR : Bit Set/Reset

I/O Mode: Input/Output mode

BSR mode does Set/Reset in port C.

I/O Mode is further divided into 3 modes:

- Mode 0: Simple I/O
- Mode 1: Handshake
- Mode 2: Bidirectional

Sweet Bird

Subject:.....
Date:.....Time:.....

Function of Pins

 # Data bus (D_0-D_7):

The 8-bit bidirectional data buses are connected to 8086 data bus for transferring data.

 # $\overline{CS}$: (Chip Select)

This is active low signal. When it is low, then data is transferred from 8086.

 # $\overline{RD}$: When it is low, read operation will be started.

 # $\overline{WR}$: When it is low, write operation will be started.

 # Address (A_0-A_1):

This is used to select the port of 8255 PPI like this:

A_1	A_0	Select
0	0	Port A
0	1	Port B
1	0	Port C
1	1	Control Register

Sweet Bird

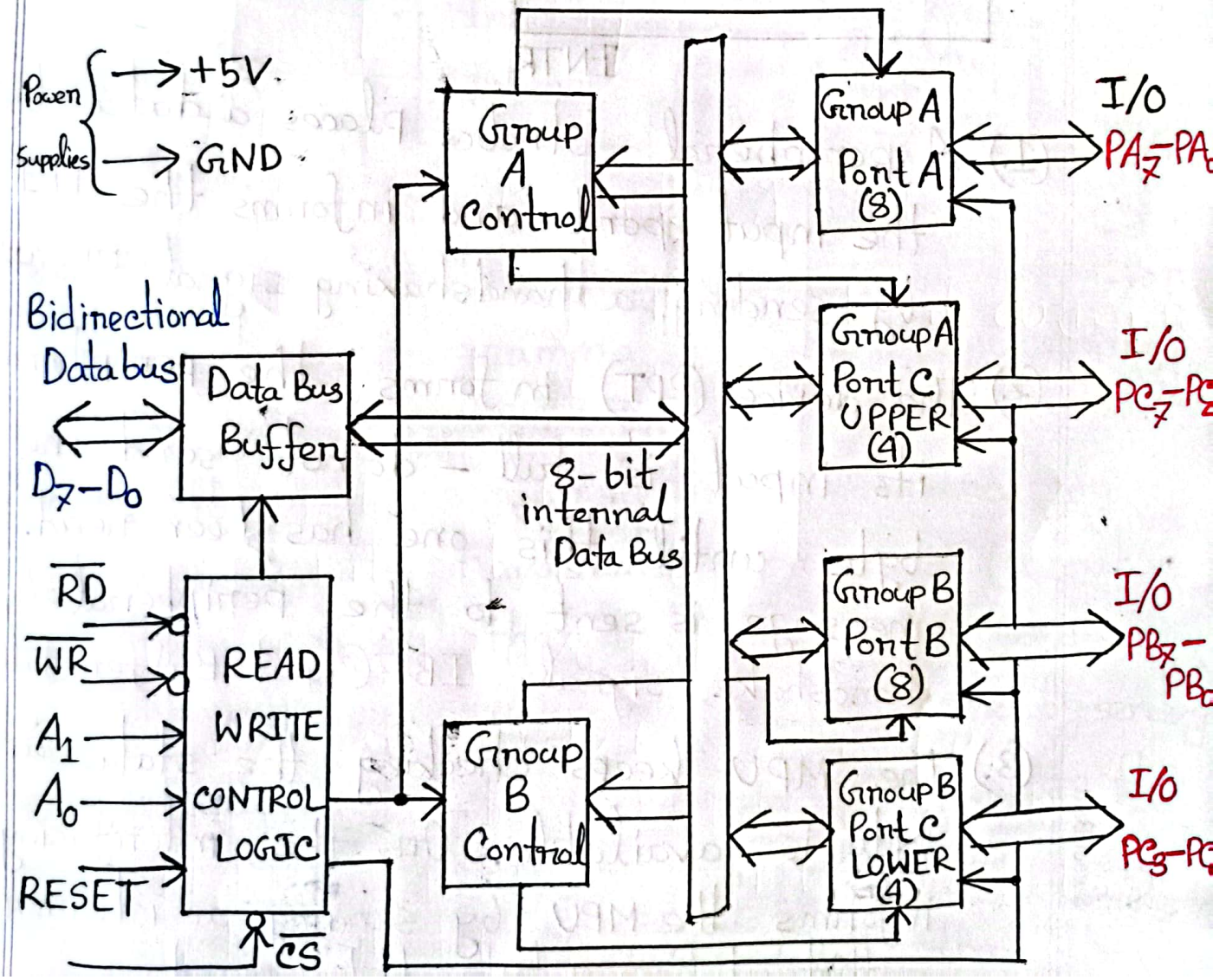
Subject:.....
Date:.....Time:.....

RESET: This is used to reset the device.
The RESET operation cleans control registers

$PA_0 - PA_7$
$PB_0 - PB_7$
$PC_0 - PC_7$
} 8-bit bidirectional data bus.

Block Diagram of 8086 with 8255 PPI

Block diagram of 8255 PPI:

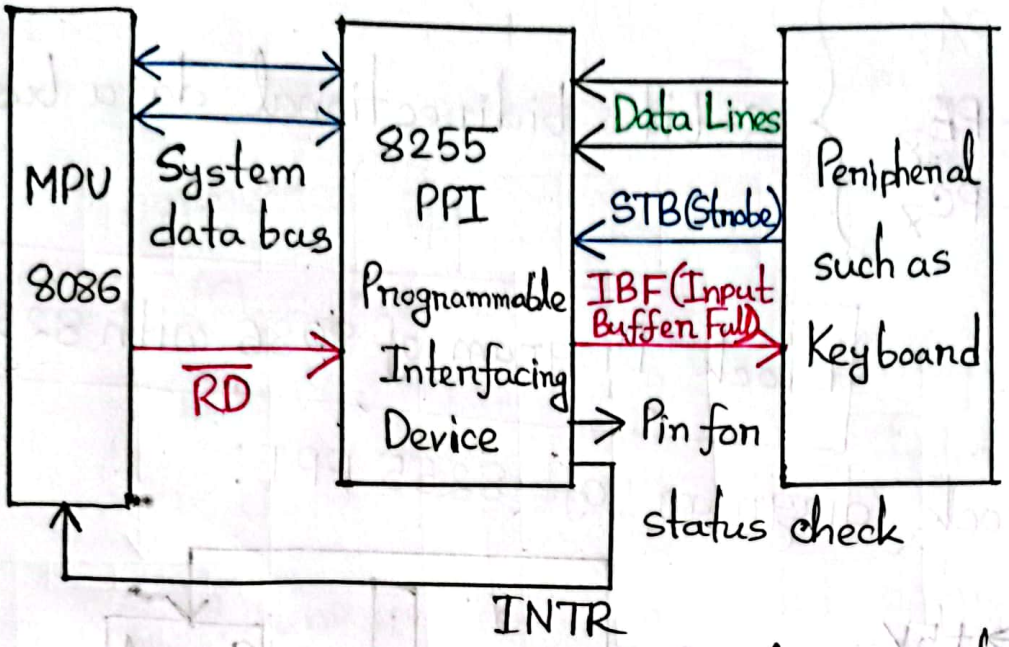


Sweet Bird

Subject:.....
Date:.....Time:.....

Data Input Interfacing with Handshaking Signal

(Read Operation)



(1.) A peripheral strobes places a data byte in the input port and informs the PPI by sending a handshaking signal STB (Strobe)

(2.) The device (PPI) informs the peripheral that its input is full – do not send the next byte until this one has been read. This message is sent to the peripheral by sending handshake signal IBF (Input Buffer full).

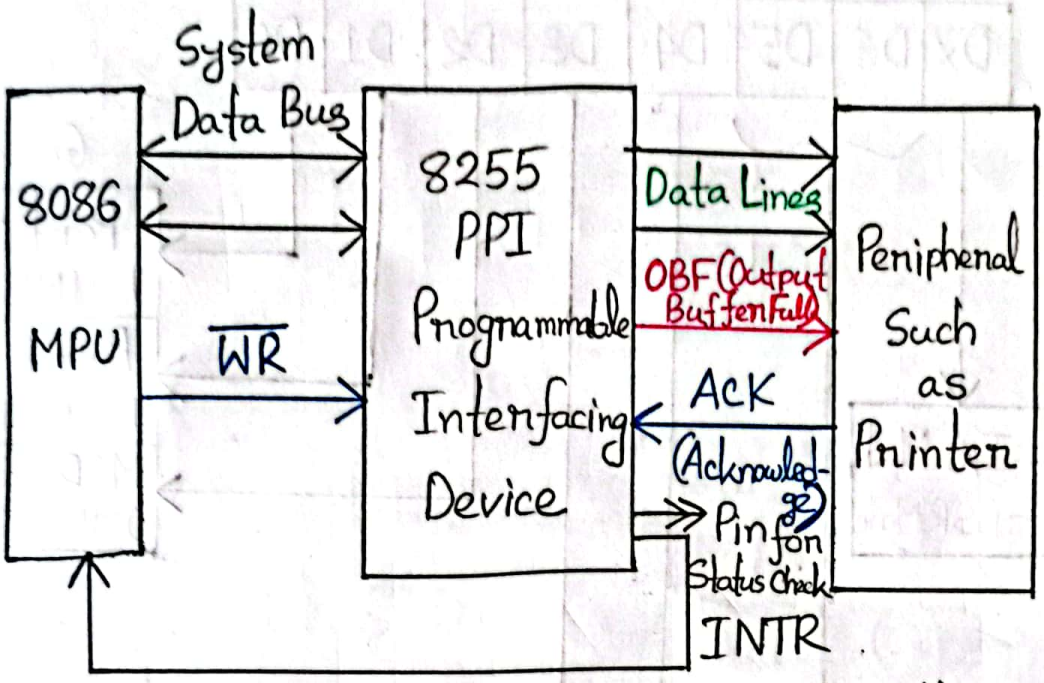
(3.) The MPU keeps checking the status until a byte is available. On, the interfacing device informs the MPU by sending an interrupt, that it has a byte to be read.

Sweet Bird

Subject:.....
Date:.....Time:.....

(4.) The MPU reads the byte by sending control signal $\overline{RD}$.

Data Output Interfacing with Handshake Signals



(1.) The MPU writes a byte into the output port of the programmable device by sending a control signal $\overline{WR}$.

(2.) The PPI informs the peripheral, by sending the handshake signal OBF (Output Buffer Full), that a byte is to be written.

(3.) The peripheral acknowledges the byte by sending back the ACK (Acknowledgement) signal to the interfacing device.

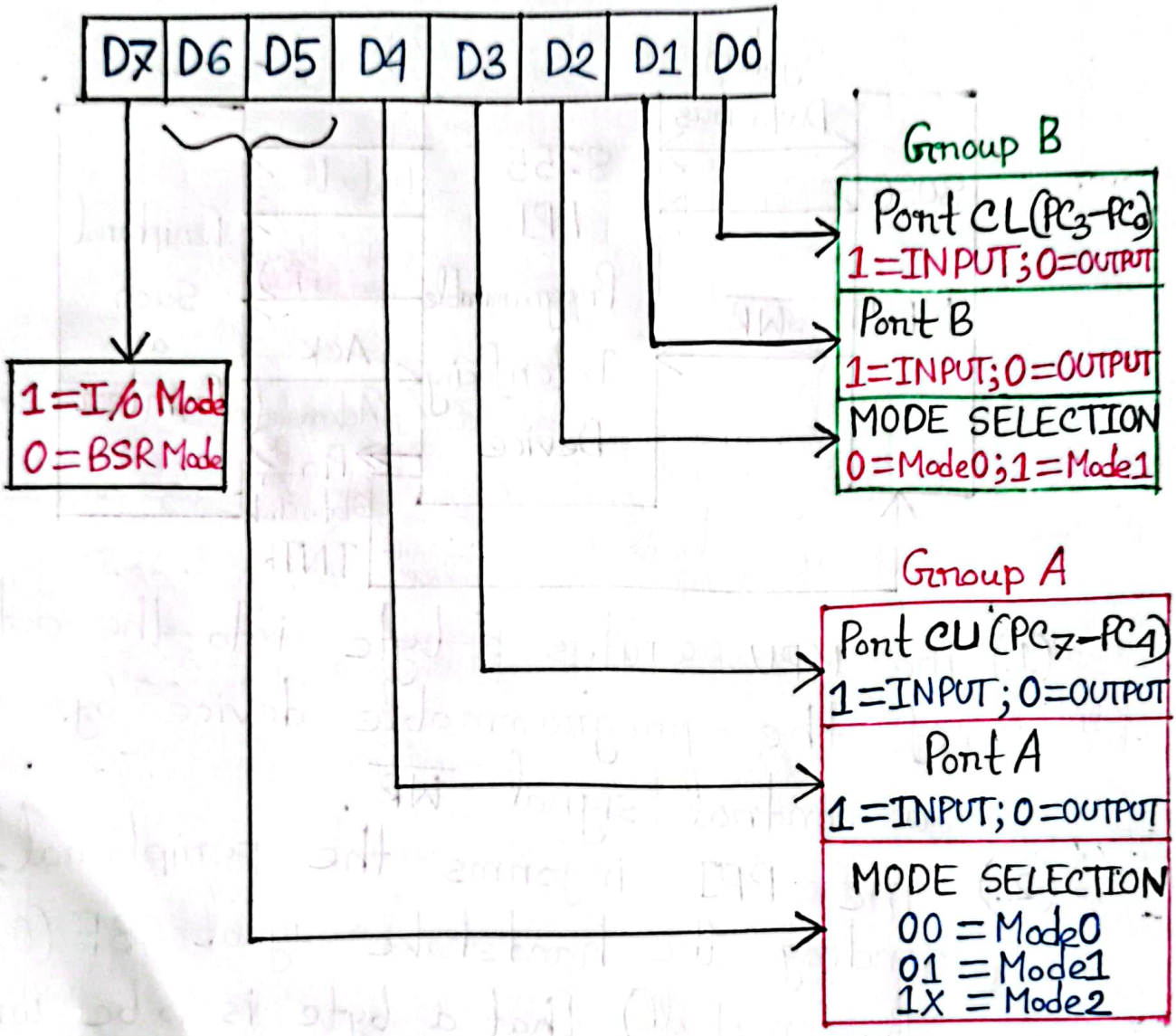
(4.) The PPI asks for the next byte through interrupting the MPU or the MPU finds out that the byte

Sweet Bird

Subject:.....
Date:.....Time:.....

has been acknowledged through the status check.

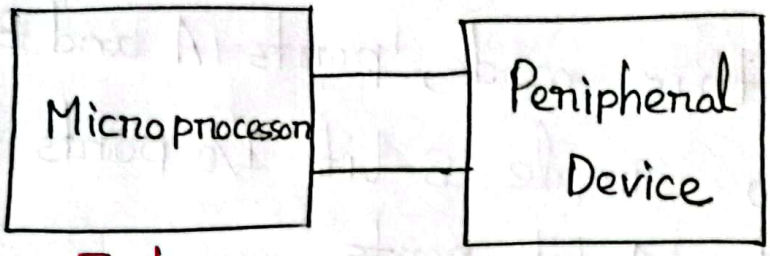
Control Word of 8255 PPI



Subject:.....
Date:.....Time:.....

Why PPI are used with 8086 microprocessor?

→



Fasten Slower

Since the 8086 microprocessor responses faster than the peripheral device, there is a chance of data overwritting. To solve this problem, we need to add a synchronizing device between the MP and the peripheral device which is called Programmable Peripheral Interface (PPI).

What is handshaking signal?

→ The microprocessor and the peripherals operate at different speeds. Therefore, signals are exchanged before data transfer between the fast-responding MPU and slow-responding peripherals. These signals are called handshaking signals.

Subject:.....
Date:.....Time:.....

Mode 0: Simple Input or Output

→ In this mode, ports A and B are used as two simple 8-bit I/O ports and port C as two 4-bit ports.

→ Each port can be programmed to function as simply an input port or an output port.

→ The input/output features in Mode0 are as follows:

(1.) Outputs are latched.

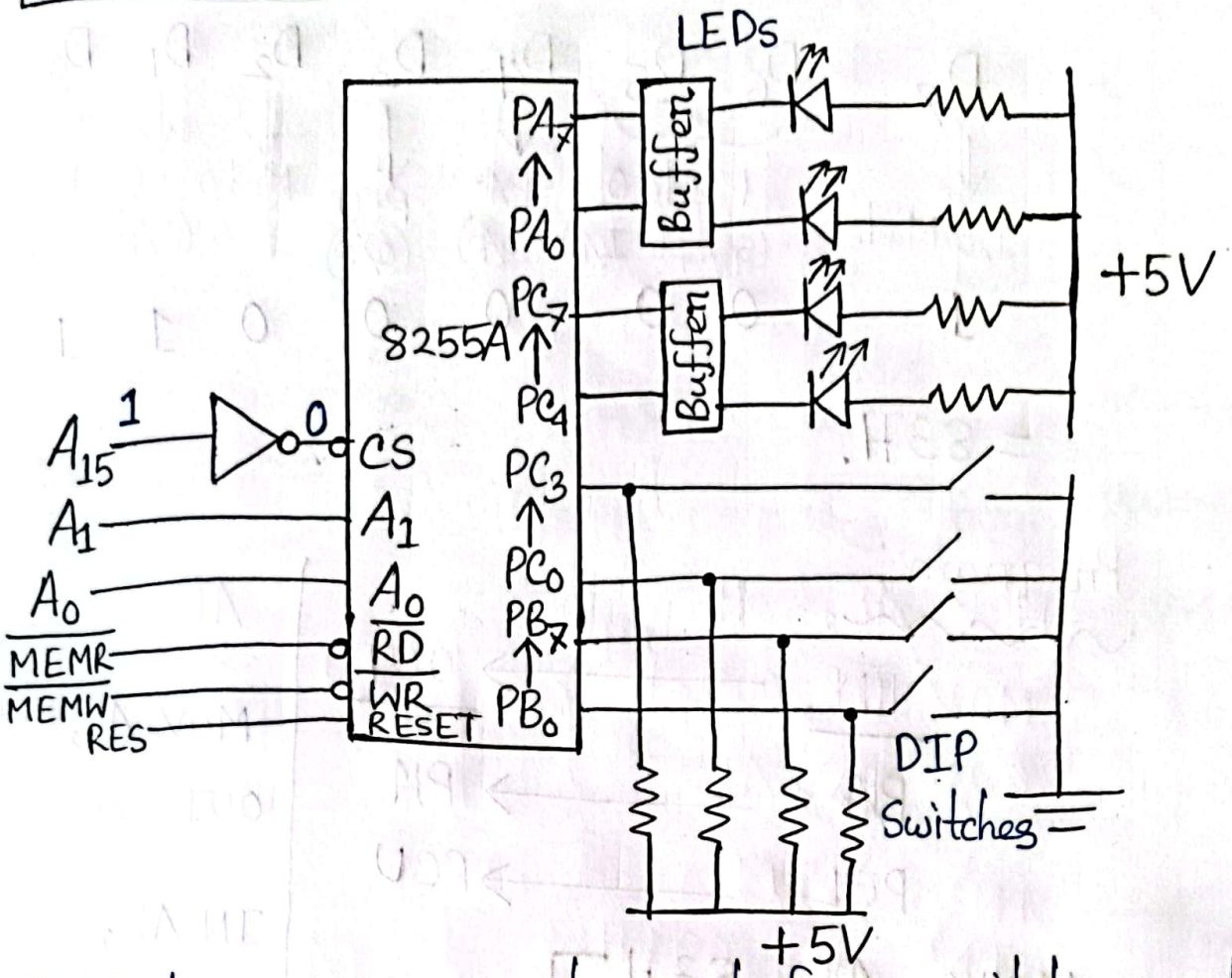
(2.) Inputs are not latched.

(3.) Ports don't have handshake or interrupt capability.

Subject:.....

Date:.....Time:.....

Problem - 01:



Write a program to read from switches and display the reading from port B to port A and from port PCL to PCU.

Control Address:

	A_{15}	A_{14}	A_{13}	.....	A_2	A_1	A_0	
Port A →	1	0	0	.....	0	0	0	= 8000 H
Port B →	1	0	0	.....	0	0	1	= 8001 H
Port C →	1	0	0	.....	0	1	0	= 8002 H
Control Register →	1	0	0	.....	0	1	1	= 8003 H.

Sweet Bird

Subject:.....
Date:.....Time:.....

Control Word:

D_7	D_6 D_5	D_4	D_3	D_2	D_1	D_0
I/O Mode	Mode 0 (Simple I/O)	PA (O/P)	PCU (O/P)	Mode 0	PB (I/P)	PCL (I/P)
1	0 0	0	0	0	1	1

$= 83H.$

Program:

IN → OUT

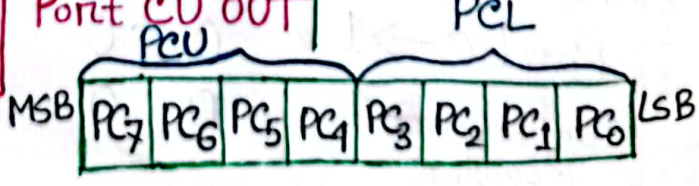
PB → PA

PCL → PCU

```
MOV  AL, 83H   ┐
MOV  DX, 8003H │ CW, CR
OUT  DX, AL    ┘

IN   AL, 8001H ┐ Port B IN
OUT  8000H, AL ┘ Port A OUT

IN   AL, 8002H ┐
SHL  AL, 04    │ Port CL IN
OUT  8002H, AL ┘ Port CU OUT

END
```

Alternative Way

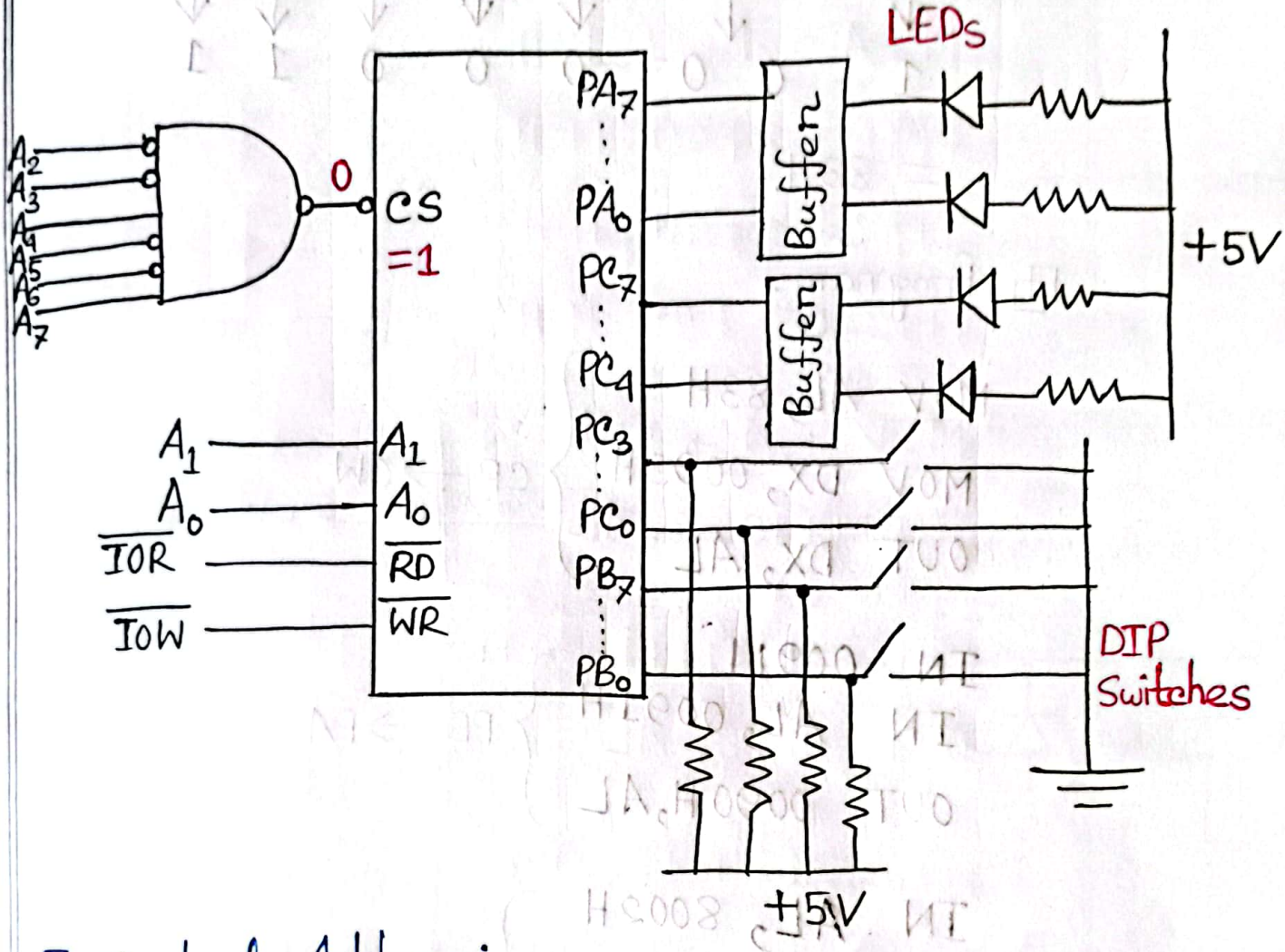
```
MOV AL, 83H
OUT 8003H, AL

IN AL, 8001H
OUT 8000H, AL

IN AL, 8002H
AND AL, 0FH
MOV CL, 04H
ROL AL, CL
OUT 8002H, AL
```



Sweet Bird

Subject:.....
Date:.....Time:.....

Problem-02: Write a program to read from switches and display the reading from port B to port A and port PCL to PCU.



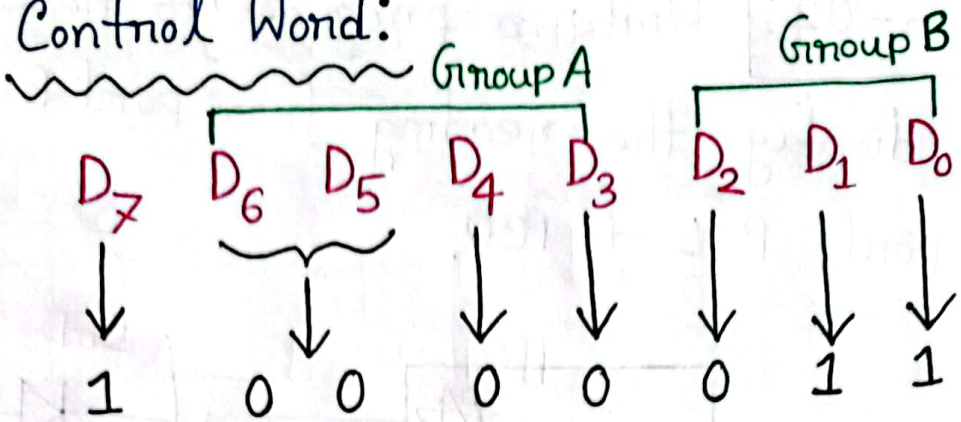
Control Address:

	A_{15} ..... A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	
Port A → 0	 0	1	0	0	1	0	0	0	0	= 0090 H
Port B → 0	 0	1	0	0	1	0	0	0	1	= 0091 H
Port C → 0	 0	1	0	0	1	0	0	1	0	= 0092 H
CR → 0	 0	1	0	0	1	0	0	1	1	= 0093 H

Sweet Bird

Subject:.....
Date:.....Time:.....

Control Word:



= 83H.

Program:

```
MOV AL, 83H
MOV DX, 0093H
OUT DX, AL
```
} CR → CW

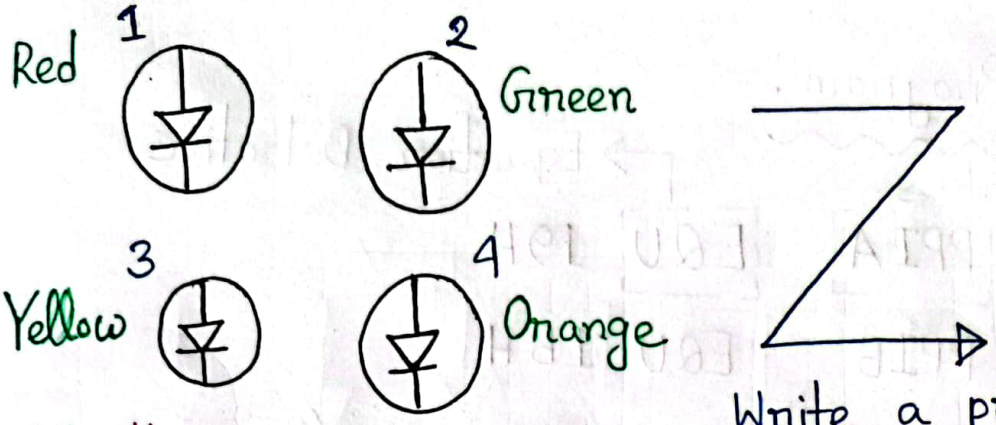
```
IN  AL, 0091H
OUT 0090H, AL
```
} PB → PA

```
IN  AL, 8002H
SHL AL, 04
OUT 8002H, AL

END
```
} PCL → PCU

Sweet Bird

Subject:.....
Date:.....Time:.....

Interfacing with LED



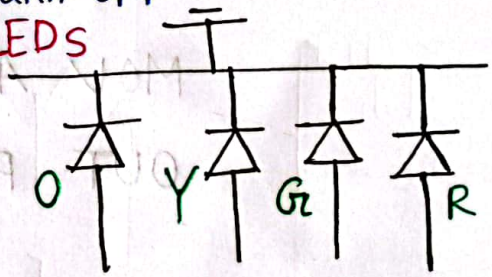
All the 4 LEDs are in Common Cathode connection.

Write a progrnam to turn-ON the 4 LEDs in this sequential orden.

$PB_0 - PB_3 \rightarrow$ LEDs are connected

$PB_4 - PB_7 \rightarrow$ Steppen Motor driven (Common Anode)

All ports are set to 1 for turn-OFF LEDs



For Red: 0 0 0 1
For Green: 0 0 1 0
For Yellow: 0 1 0 0
For Orange: 1 0 0 0

Solution:

Control Address:

Address are given as:

PA = 19 H
PB = 1B H
PC = 1D H
CR = 1FH

Will be given in the question

Control Word: Here, Port A & Port C are don't care, since they are not used.

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
↓	↓	↓	↓	↓	↓	↓	↓
80H = 1	0	0	0	0	0	0	0

Control Word = 80H

Sweet Bird

Subject:.....
Date:.....Time:.....

∴ Control Word = 80H.

Program:

Variables → Equation Detective

```
PPIA    EQU 19H
PPIB    EQU 1BH
PPIC    EQU 1DH
PPIC_C  EQU 1FH

MOV  AL, 80H
OUT  PPIC_C, AL        ; CR → CW
                       ; Steppen  LEDs
MOV  AL, 11110001B     ; Red LED ON
OUT  PPIB, AL          ; Stepper Motor OFF

MOV  AL, 11110010B     ; Green LED ON
OUT  PPIB, AL

MOV  AL, 11110100B     ; Yellow LED ON
OUT  PPIB, AL

MOV  AL, 11111000B     ; Orange LED ON
OUT  PPIB, AL
END
```

Sweet Bird

Subject:.....
Date:.....Time:.....

Program for 4 LED Blinking:

If the 4 LED blinking are to be done for infinite number of times:

```
PPIA   EQU 19H
PPIB   EQU 1BH
PPIC   EQU 1DH
PPIC_C EQU 1FH

XXX: MOV CX,04H
     MOV AL,11110001B ; Starts from the RED Led
ABC: OUT PPIB,AL
     SHL AL,1 ; Shifted for sequentially writing other LEDs
     OR AL,11110000B ↑ To keep the steppen moton off after SHL (Shift Left)
     DEC CX
     JNZ ABC
     CMP CX,0000H
     JZ XXX
```

Sweet Bird

Subject:.....
Date:.....Time:.....

To blink the 4 LED sequence for 10 times,

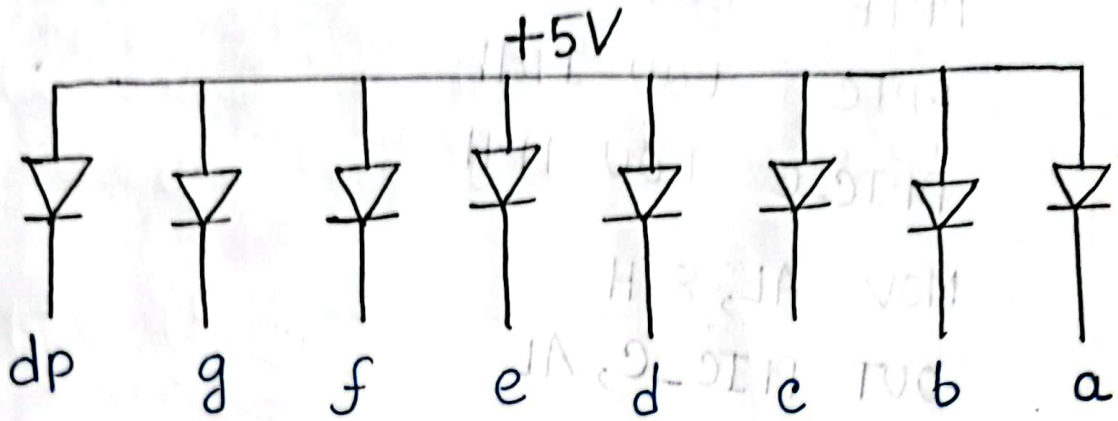
```
MOV  DX,10
•WHILE  DX>0
XXX: MOV CX,4
     DEC DX
     MOV AL,1111 0001 B
ABC:OUT  PPIB, AL
     SHL  AL,1
     OR  AL, 11110000B
     DEC CX
     JNZ ABC
    CMP CX, 0000H
    JE XXX
  •ENDW
```

Subject:.....
Date:.....Time:.....

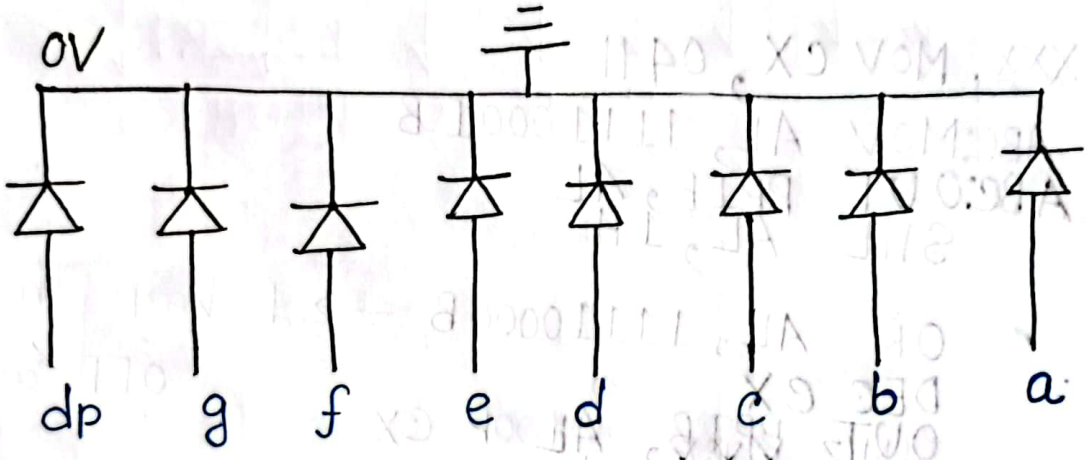
7-Segment Display

Two Types connection:

(i.) Common Anode:



(ii.) Common Cathode:



- To turn ON in common anode connection:
  Logic 0 will be given into cathode (dp g ..... a)

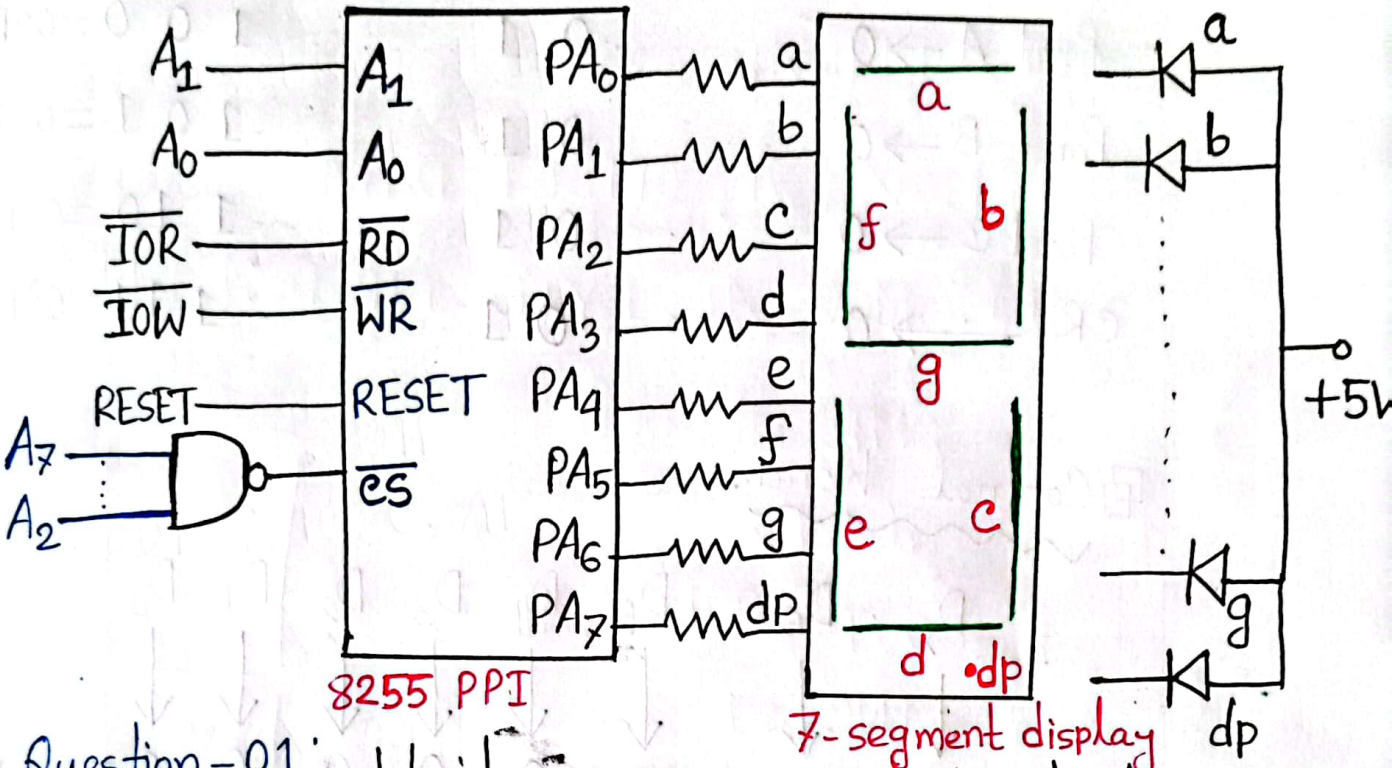
- To turn ON in common cathode connection:
  Logic 1 will be given into anode
  (dp g f ..... a)

Decimal Pointer (dp) will always be set to = 0 (don't care)

Sweet Bird

Subject:.....
Date:.....Time:.....

Problem :



Question-01: Write a program to display 0-9 digits using 7-segment display.

Question-02: Write a program to display a specific digit 1/2/3 .....

Question-03: To display a specific alphabet A/B/C/D.

01 Ans.:

Control Address:

Here, $A_{15}, A_{14}, \ldots\ldots A_8$ are don't care on 0.

Sweet Bird

Subject:.....
Date:.....Time:.....

∴ Control Addresses : To keep $\overline{CS}$ = 0

A_{15} ..... A_8 A_7 ..... A_2 A_1 A_0

Port A → 0 ..... 0, 1 ..... 1 0 0 = 00FCH

Port B → 0 ..... 0 1 ..... 1 0 1 = 00FDH

Port C → 0 ..... 0 1 ..... 1 1 0 = 00FEH

CR → 0 ..... 0 1 ..... 1 1 1 = 00FFH

Control Word :

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	
1	0	0	0	0	0	0	0	= 80 H
I/O Mode	Group A Port A Mode 0 : 00		Port A O/P	PCU don't care	Port B Mode 0	Port B don't care	PCL don't care	

If we have to show any data sequentially, then we must keep the data sequentially to a memory map/memory segment.

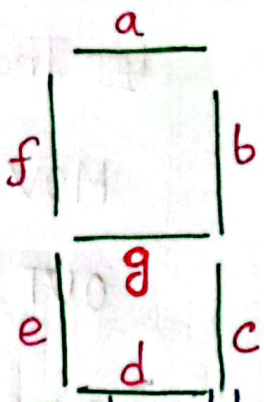
After that, we will take the data one by one by increasing the offset address of the memory segment.

Sweet Bird

Subject:.....
Date:.....Time:.....

Program:

```
MOV AL, 80H
OUT 00FFH, AL
```
CR → CW




```
L2: MOV SI, OFFSET DATA;
```
→ Copy offset address of label DATA to the SI register

```
L1: MOV AL, BYTE PTR DS:[SI]
    CMP AL, 00H ; AL-00H
    JZ L2 ; will jump to Label L2 if Z=1
```
or, AL-00H = 00H
```
    OUT FCH, AL
    INC SI ; SI = SI+1
    JMP L1 ; Jump to L1 for showing next Byte
```

```
DATA: DB C0H
      DB F9H
      DB A4H
      DB B0H
      DB 99H
      DB 92H
      DB 82H
      DB F8H
      DB 80H
      DB 90H
```
Define Byte

	DS : SI	
0 →	C0	1000H
1 →	F9	1001H
2 →	A4	1002H
3 →	B0	1003H
4 →	99	1004H
5 →	92	1005H
6 →	82	1006H
7 →	F8	1007H
8 →	80	1008H
9 →	90	1009H
	(00) Blank	100AH

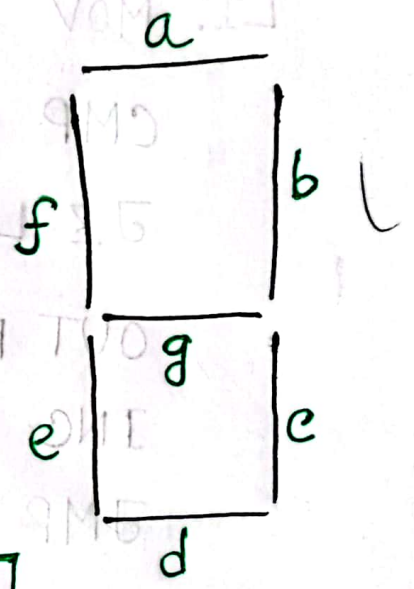
For this case, AL is CMP with 00H

Sweet Bird

Subject:.....
Date:.....Time:.....

The Program can also be written as:

```
    MOV AL, 80H
    OUT 00FFH, AL
L2: MOV SI, OFFSET DATA
L1: MOV AL, BYTE PTR DS:[SI]
    CMP AL, 00H
    JZ L2
    OUT 00FCH, AL
    INC SI
    JMP L1
DATA: DB 11000000B
      DB 11111001B
      DB 10100100B
      DB 10110000B
      DB 10011001B
      DB 10010010B
      DB 10000010B
      DB 11111000B
      DB 10000000B
      DB 10010000B
```



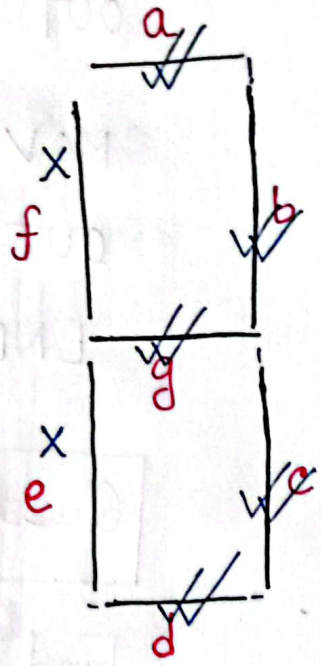
Sequence for 0-9 Digits

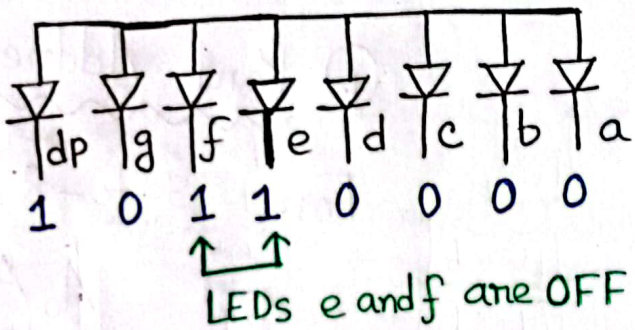
Sweet Bird

Subject:.....
Date:.....Time:.....

(02) Ans.: To display digit 3 only in 7-segment

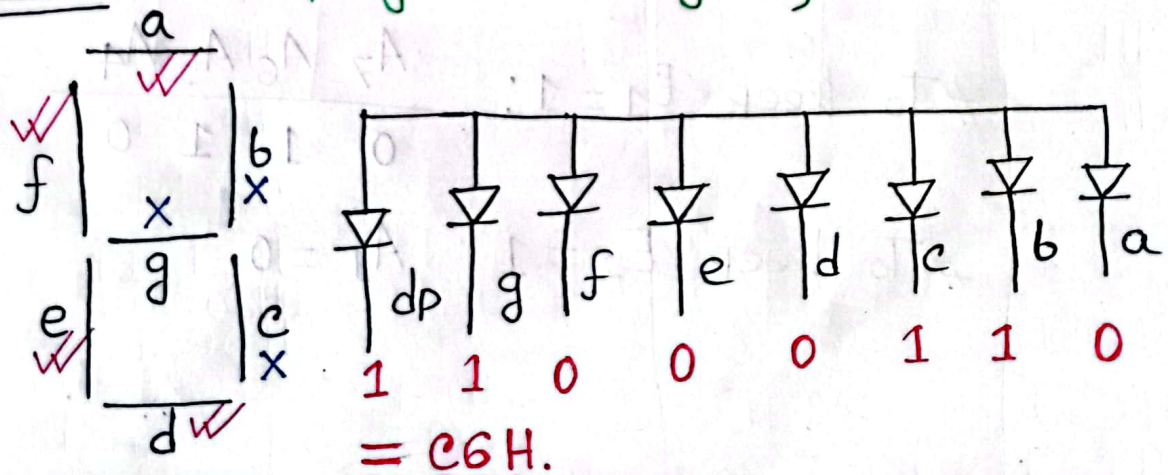
```
PPIA    EQU  00FCH
PPIB    EQU  00FDH
PPIC    EQU  00FEH
PPIC_C  EQU  00FFH

MOV   AL, 80H
OUT   PPIC_C, AL

MOV   AL, B0H
OUT   PPIA, AL
END
```





The Data is = 1011 0000 B

(03) Ans.: To Display C in 7-segment, = B0H



= C6H.

Sweet Bird

Subject:.....
Date:.....Time:.....

Program:

```
MOV  AL, 80H
OUT  00FFH, AL
```
] CR → CW

```
MOV  AL, C6H
OUT  00FCH, AL
```
] To display C

```
END
```

LED/7-segment Interfacing with 8086

Question 6(c.) Question'15

Find the following :-

(i.) Port Address:

For 3-to-8 line decoder-

A_0 A_3 A_2	Output
1 0 0	Y_4 → For 7-Segment 1
0 1 1	Y_3 → For 7-Segment 2

#To keep $E_1 = 1$:

A_7	A_6	A_5	A_4
0	1	1	0

#To keep $E_2 = 1$: $A_1 = 0$

Sweet Bird

Subject:.....
Date:.....Time:.....

Port Address of 7-Segment 1:

A_{19} ..... A_8 A_7 A_6 A_5 A_4 A_3 A_2 A_1 A_0

0 ..... 0 0 1 1 0 0 0 0 1

$= 00061H$.

Segment Address = 0000H

Offset Address = 0061H.

Port Address of 7-Segment 2:

A_{19} ..... A_8 A_7 A_6 A_5 A_4 A_3 A_2 A_1 A_0

0 ..... 0 0 1 1 0 1 1 0 0

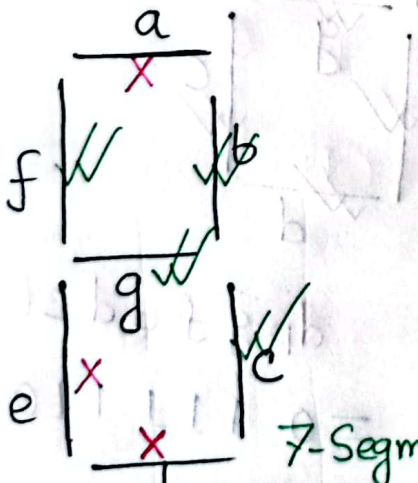
$= 0006CH$.

Segment Address = 0000H

Offset Address = 006CH.

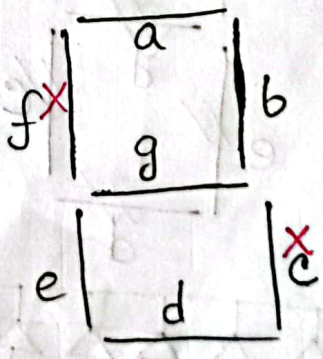
(ii) Write instructions to display the number "42" and "86" at common-cathode 7-segment LED port.

Displaying "42" in the 7-segment:



7-Segment 1

Data = 01100110B



7-Segment 2

Data = 01011011B

Both 7-segment are Common Cathode

Sweet Bird

Subject:.....
Date:.....Time:.....

Program to show "42":

```
MOV  AX, 0000H
MOV  DS, 0000H
MOV  AL, 01100110B

MOV  BX, 0061H
OUT  BX, AL
```

To display "4"

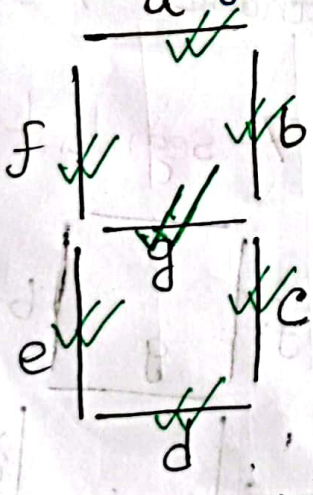
Not necessary

```
MOV  AX, 0000H
MOV  DS, AX
MOV  AL, 01011011B
MOV  BX, 006CH
OUT  BX, AL
```

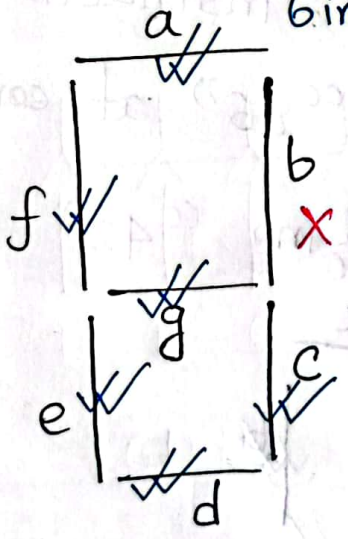
To display "2"

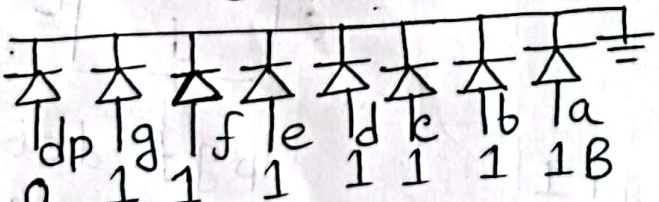
Displaying "86" in 7-Segment:

8 in 7-Segment-1



6 in 7-segment-2





dp g f e d c b a

0 1 1 1 1 1 0 1 B

= 7FH.

dp g f e d c b a

= 0 1 1 1 1 1 0 1 B

= 7DH.

Sweet Bird

Subject:.....
Date:.....Time:.....

Program:

```
MOV AL, 7FH
MOV DX, 0061H
OUT DX, AL
```

To display "8" in 7-segment 1

```
MOV AL, 7DH
MOV DX, 006CH
OUT DX, AL
```

To display "6" in 7-segment 2

Problem : 2 (c.) Question 13 - 14

A_0	A_2	A_3	
1	0	1	Y_5
0	1	1	Y_3

(i.) Port Address:

#Port Address of 7-Segment 1: (Common Anode)

A_{19}	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0
0	0	1	1	1	1	1	0	1	1

= 000FBH.

Port Address of 7-Segment 2: (Common Cathode)

A_{19}	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0
0	0	1	1	1	1	1	1	1	0

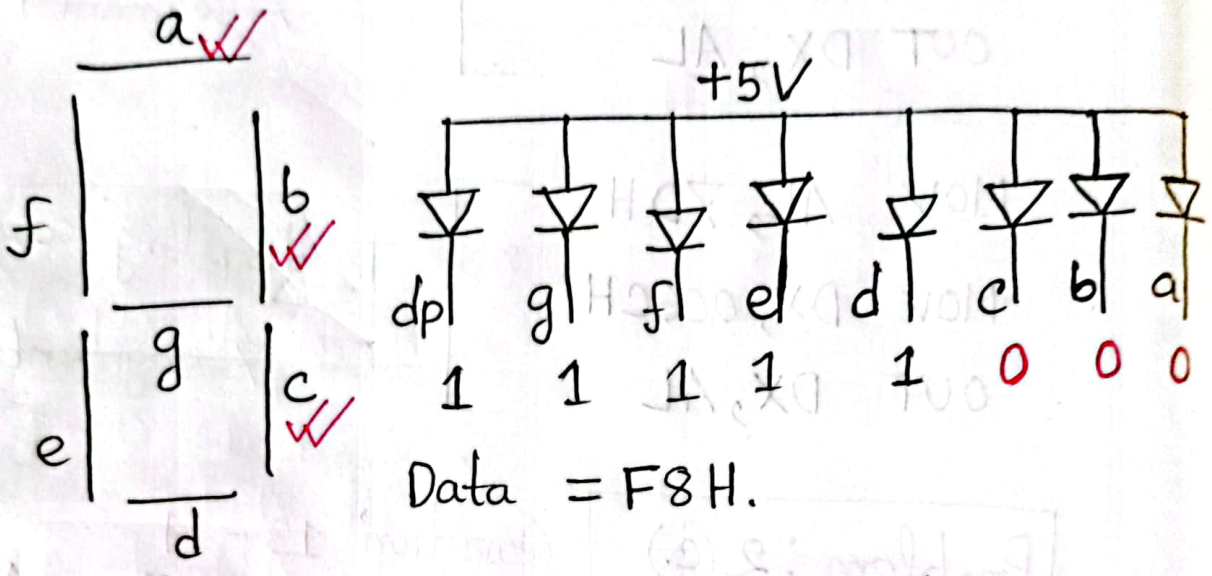
=000FEH.

Sweet Bird

Subject:.....
Date:.....Time:.....

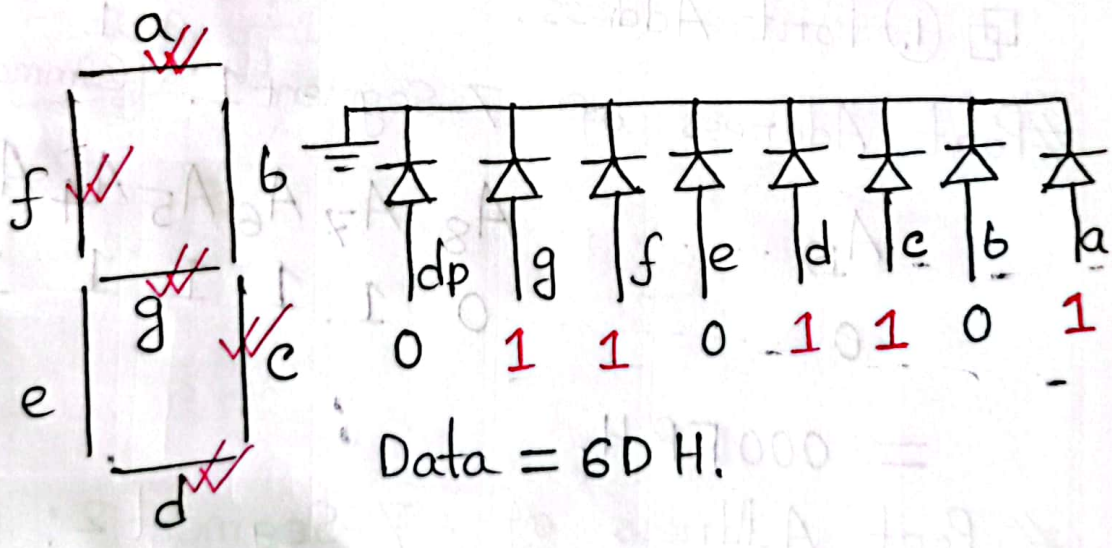
(11.) Write instructions to display the number "75" at the 7-segment LED port.

Displaying "7" at 7-segment 1 (C.A.):



Data = F8H.

Displaying "5" at 7-segment 2 (C.C):



Data = 6DH.

Sweet Bird

Subject:.....
Date:.....Time:.....

Program:

```
MOV  AL, F8H
MOV  DX, 00FBH
OUT  DX, AL
```

To display "7"

```
MOV  AL, 6DH
MOV  DX, 00FEH
OUT  DX, AL
```

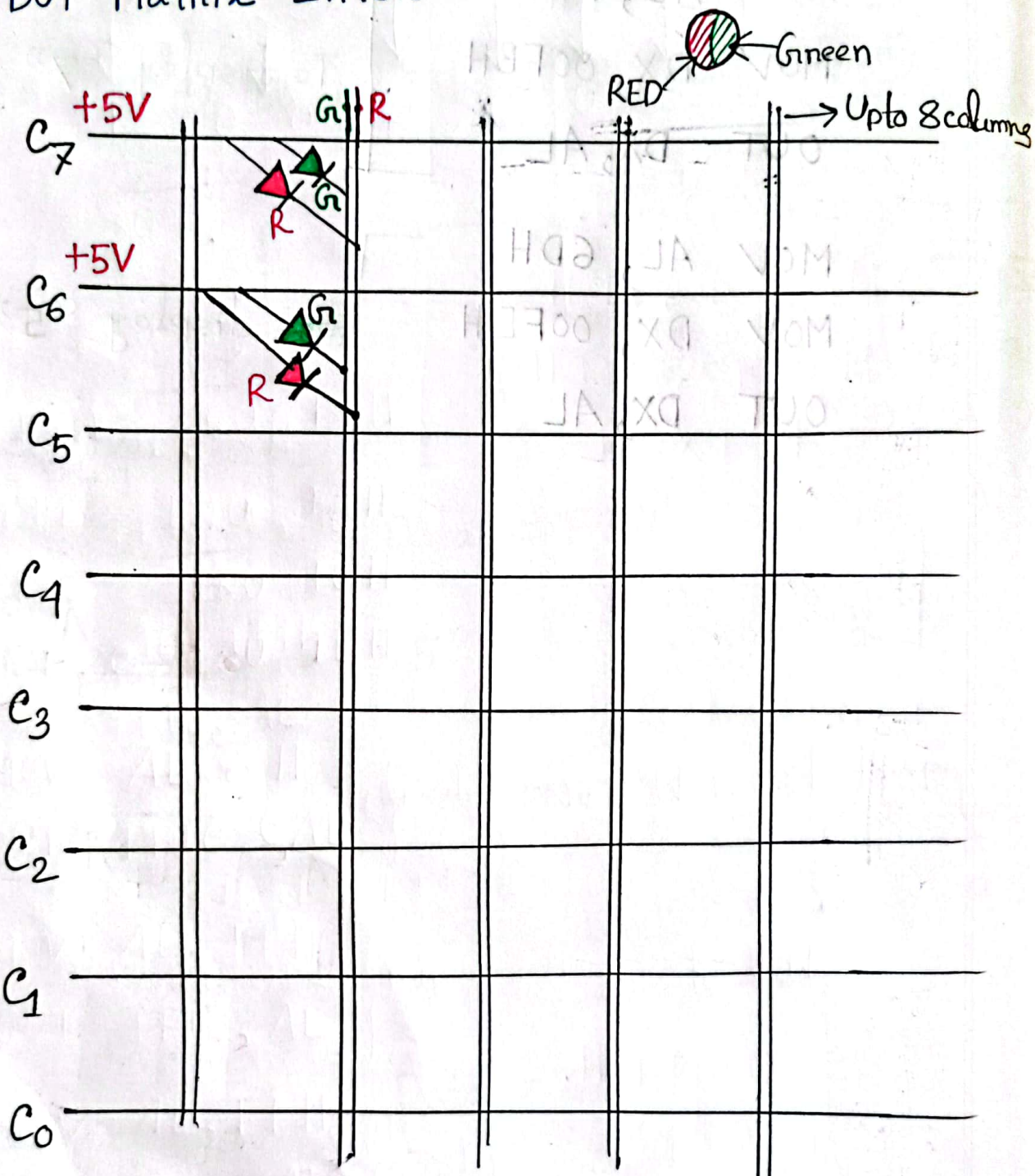
To display "5"

Subject:.....

Date:.....Time:.....

8255 PPI Interfacing with DOT Matrix

DOT Matrix Internal Structure:



PortA → PA_7 PA_6 PA_5 PA_4 PA_3 PA_2 PA_1 PA_0 : RED LED

PortB → PB_7 PB_6 PB_5 PB_4 PB_3 PB_2 PB_1 PB_0 : Green LED

PortB → Green LED Cathode

PortA → Red LED Cathode

} PortC / Both LEDs Anode.

Sweet Bird

Subject:.....
Date:.....Time:.....

- In the row, all the LED is connected in common anode.
- In the column, LED is connected in common-cathode.

Considering Common-Anode Configuration:

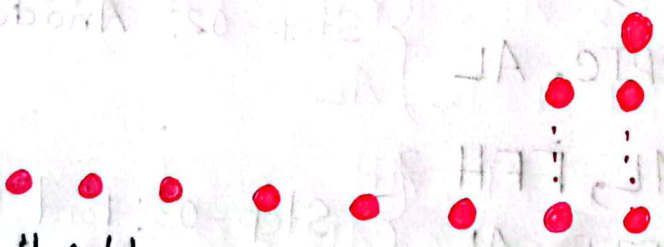
To turn ON all the LEDs,

PA_7 ---- $PA_0 = 0$
PB_7 ---- $PB_0 = 0$
} According to requirement

To turn ON both Red and Green LEDs,

PC_7 ---- $PC_0 = 1$

Question: Write a program to sequentially generate the 1st row of DOT Matrix with Red LED.



Given, Port Address:

PA → 18H PB → 1AH
PC → 1AH CR → 1EH

Sweet Bird

Subject:.....
Date:.....Time:.....

Control Word:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	0	0	0	0	0

$= 80\ H.$

Program:

```
PPIA    EQU  18H
PPIB    EQU  1AH
PPIC    EQU  1CH
PPIC_C  EQU  1EH

MOV  AL,80H
OUT  PPIC_C,AL      } Step-01: CW is set to CR

MOV  AL, FFH
OUT  PPIC, AL       } Step-02: Anode ON

MOV  AL, FFH
OUT  PPIB, AL       } Step-03: Pont B OFF, Gneen LED OFF

L1: MOV  AL, 11111110B
L2: OUT  PPIA, AL
    SHL  AL,1
```

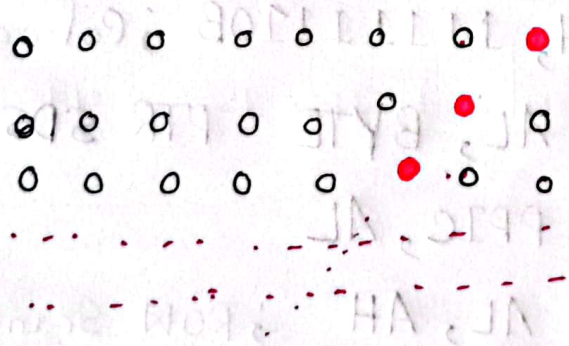
Sweet Bird

```
CMP  AL,00H
JZ L1
JMP L2
```

Prognam for generating one point (one LED in a row) sequentially at a time

```
L1: MOV  AL, 11111110B
L2: OUT  PPIA, AL
    ROL  AL,1
    CMP  AL, 00H
    JZ L1
    JMP L2
```

Output of this prognam:



Subject:.....
Date:.....Time:.....

Problem: Write a program to show generate 'A' using RED LED in the dot matrix.

```
PPIA    EQU  18H
PPIB    EQU  1AH
PPIC    EQU  1CH
PPIC_C  EQU  1EH

MOV  AL, 80H
OUT  PPIC_C, AL

MOV  AL, FFH
OUT  PPIB, AL      ] To turn OFF Green LED

L1: MOV SI, OFFSET POINT
    MOV AH, 11111110B ; Column Scanning
L2: MOV AL, BYTE PTR DS:[SI]
    OUT PPIC, AL
    MOV AL, AH    ; ROW Scanning
    OUT PPIA, AL  ; Moving data to Pont A
    CALL TIMER ; for delay
    INC SI
    ROL AH
    JC L2  <-- Jump if carry
```

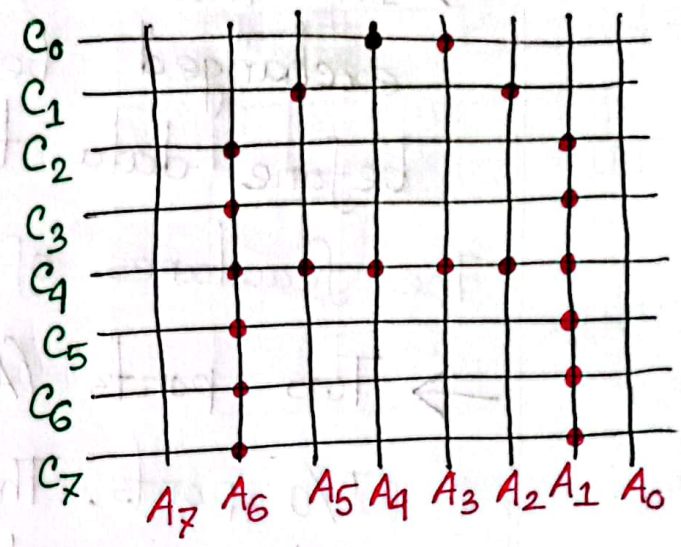
Row-wise

Rightmost → LSB

Leftmost → MSB

Sweet Bird

```
    JMP L1
    END
POINT:
    DB  00000000B
    DB  11111100B
    DB  00010010B
    DB  00100001B
    DB  00010001B
    DB  00010010B
    DB  11111100B
    DB  00000000B
```



Subject:.....
Date:.....Time:.....

Mode 1: Input or Output with Handshake

→ In this mode, handshake signals are exchanged between the MPU and peripheral before data transfer.

The feautures of this mode:

→ Two ports (A and B) function as 8-bit I/O ports. They can be configured as either input or output ports.

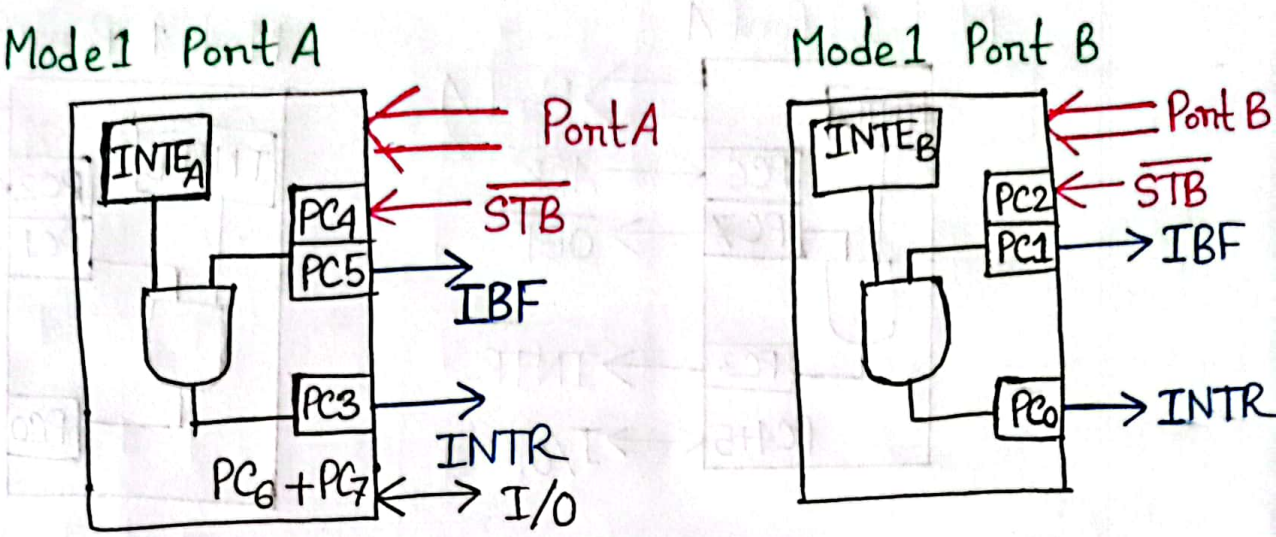
→ Each port uses three lines from port C as handshake signals. The remaining two lines of port C can be used for simple I/O operations.

Input and Output data are latched.

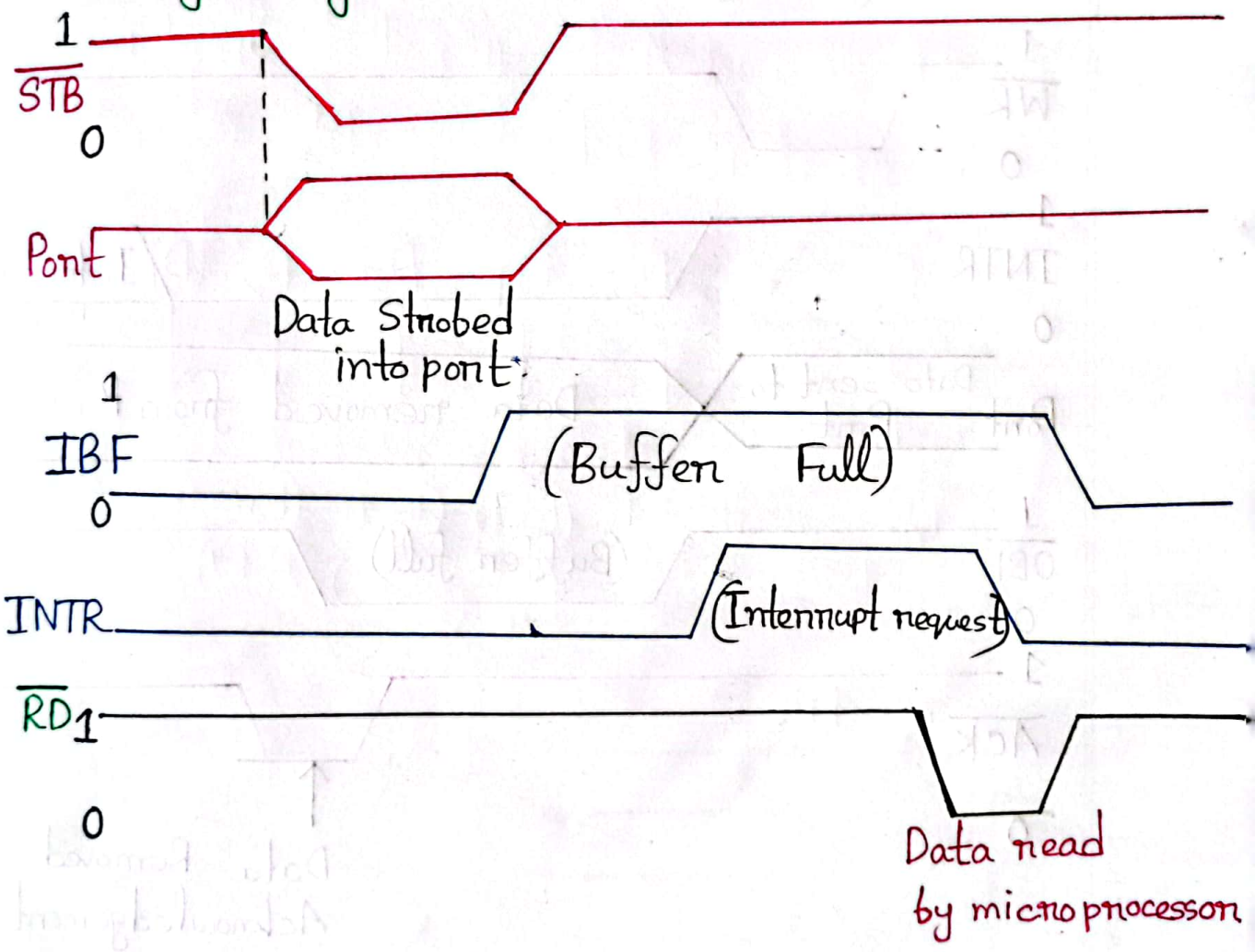
Interrupt logic is supported.

Subject:.....
Date:.....Time:.....

Input Handshaking (Read Operation) in Mode1:



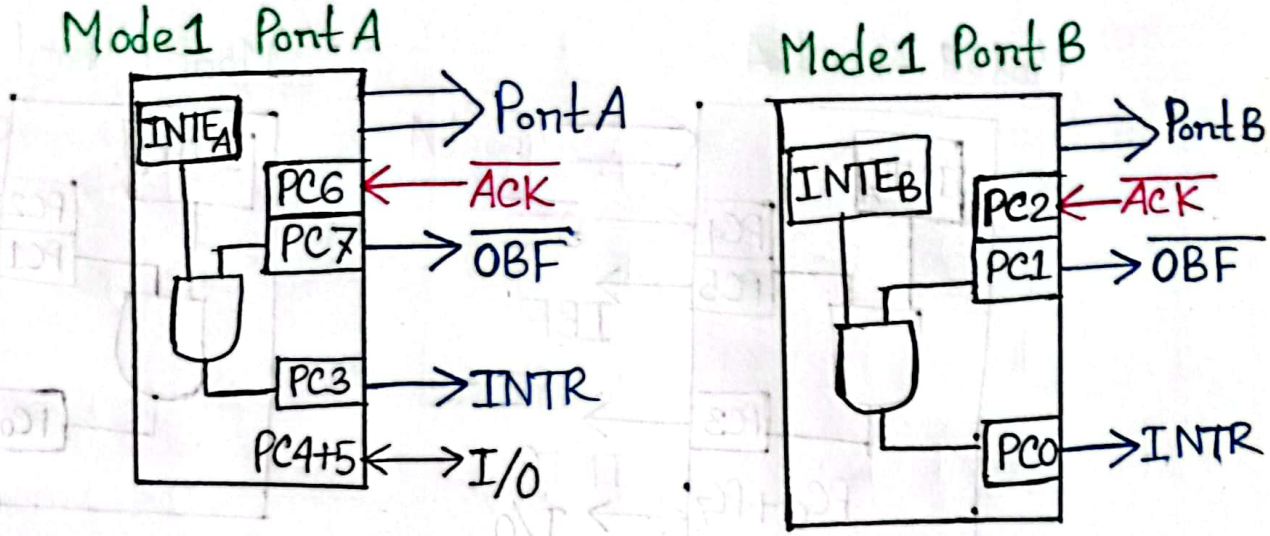
#Timing Diagram:



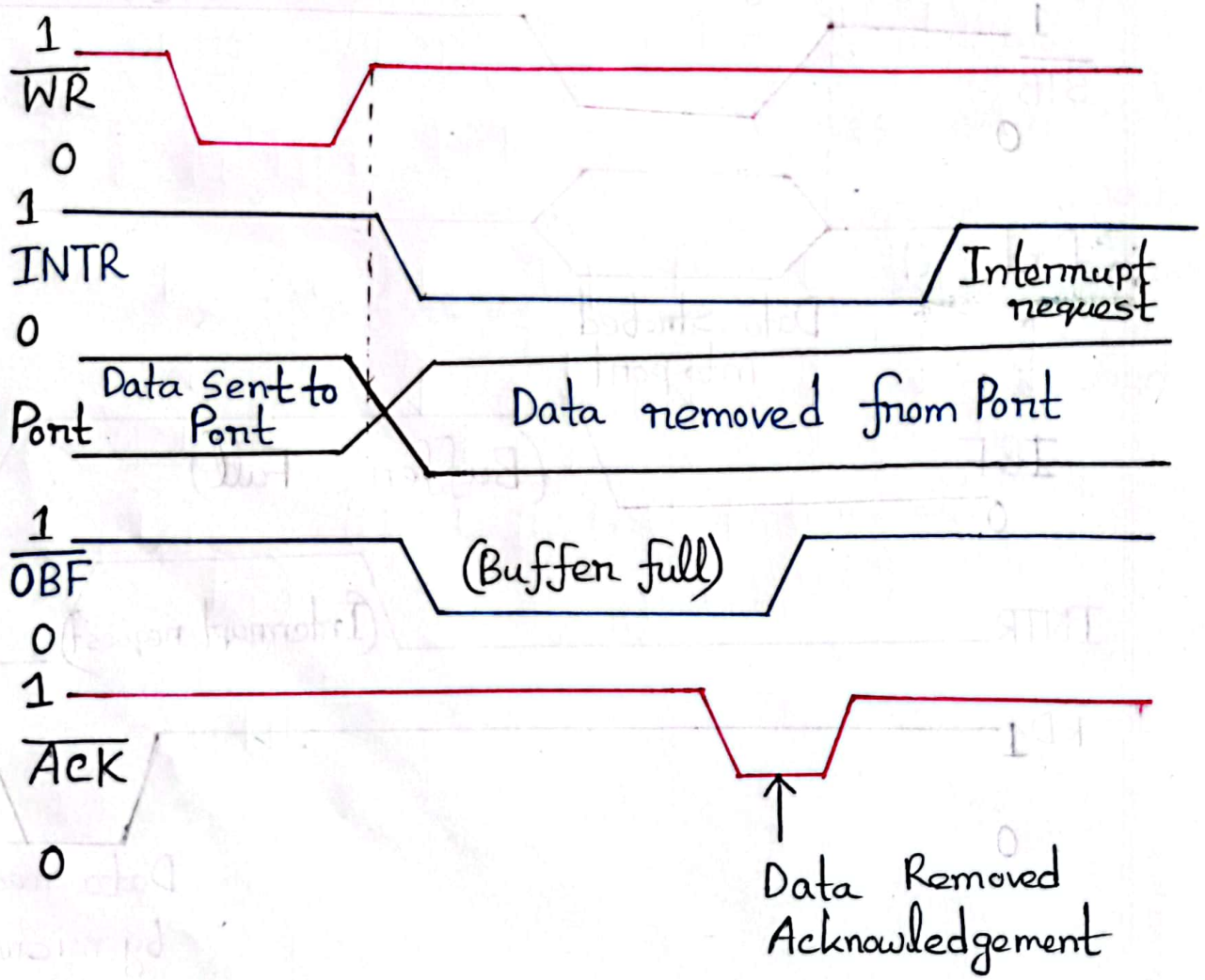
Sweet Bird

Subject:.....
Date:.....Time:.....

Output Handshaking (For Write Operation) in Mode1



Timing Diagram:

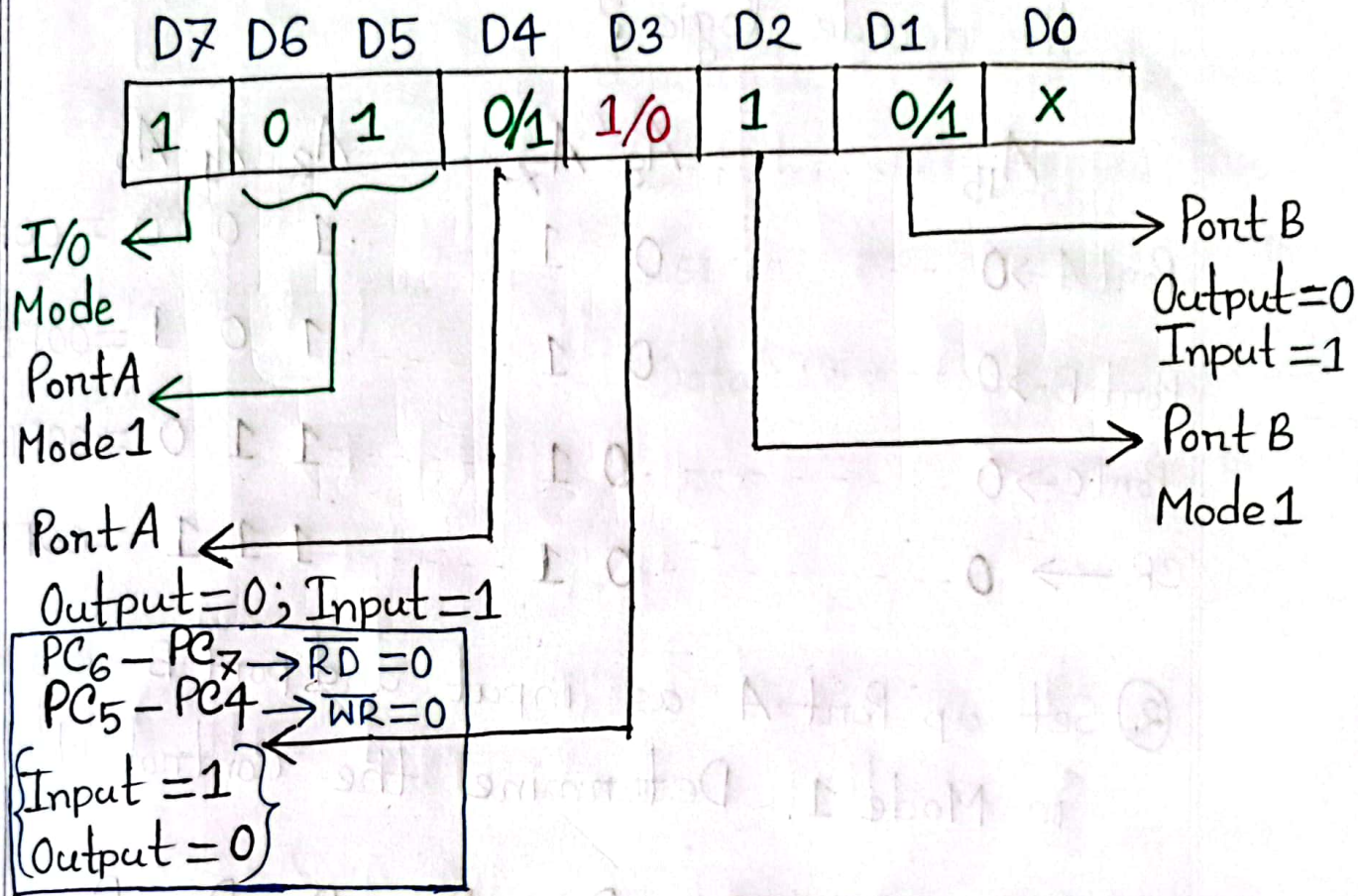


Sweet Bird

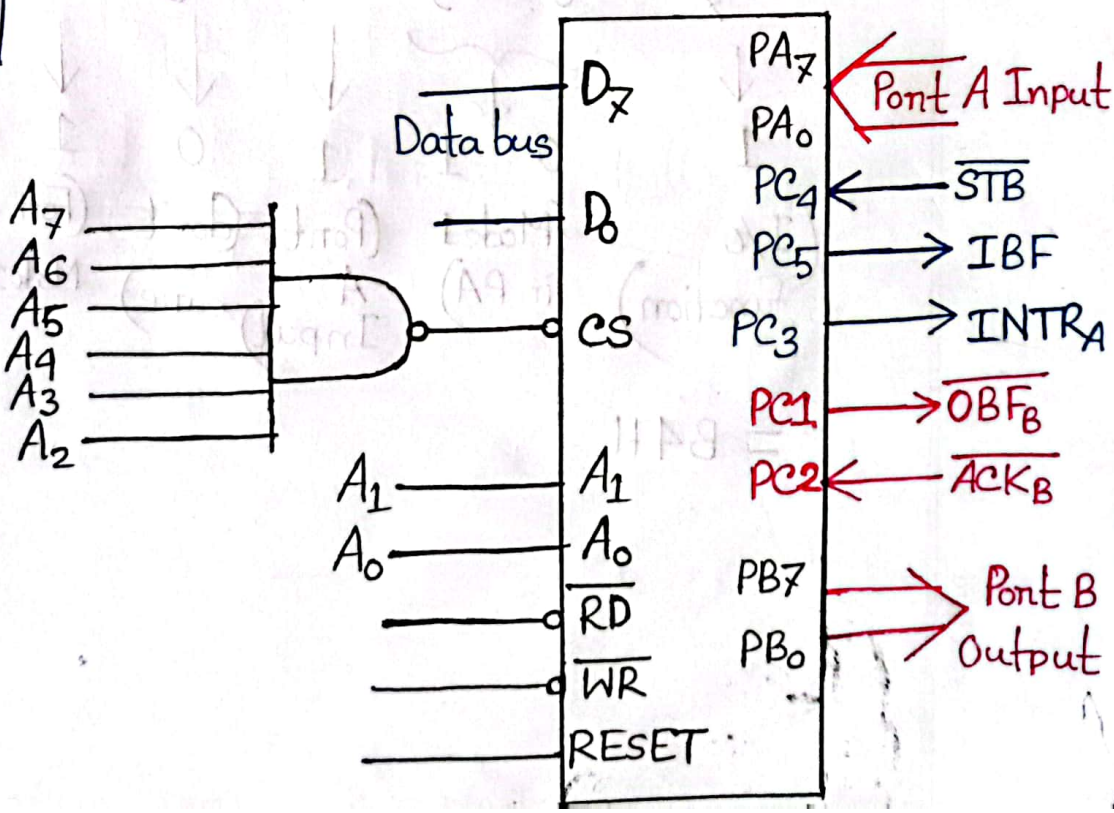
Subject:.....

Date:.....Time:.....

Control and Status Word



Problem



Sweet Bird

Subject:.....

Date:.....Time:.....

① Find the port addresses by analyzing the decode logic?

	A_{15} ..... A_8	A_7 ..... A_2	A_1	A_0	
Port A →	0 0	1 1	0	0	= 00FCH
Port B →	0 0	1 1	0	1	= 00FDH
Port C →	0 0	1 1	1	0	= 00FEH
CR →	0 0	1 1	1	1	= 00FFH

② Set up Port A as input and port B as output in Mode 1. Determine the Control Word?

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
↓	↓	↓	↓	↓	↓	↓	↓
1	0	1	1	0	1	0	0
(I/O function)	(Mode 1 in PA)		(Port A Input)	(don't care)	(For Mode 1)	(Port B output)	(don't care)

= B4H.

Sweet Bird

Subject:.....
Date:.....Time:.....

③ Determine the BSR word to enable $INTE_A$ (port A).

→ To enable or generate the signal $INTE_A$, flip-flop $INTE_A$ must be set by 1. This can be done by using the BSR mode to set $PC_4 = 1$. [For Data IN → PC_4 ($\overline{IBF}$) must be set
For Data OUT → PC_6 ($\overline{ACK}$) must be set]

The BSR word is:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
↓	↓	↓	↓	↓	↓	↓	↓
0	0	0	0	1	0	0	1 = 09H
↓				Bit PC_4			[Set Operation]
BSR Mode							

Program:

```
MOV AL, 09H
OUT 00FFH, AL
```

Sweet Bird

Subject:.....
Date:.....Time:.....

Bit Set Reset (BSR) Mode

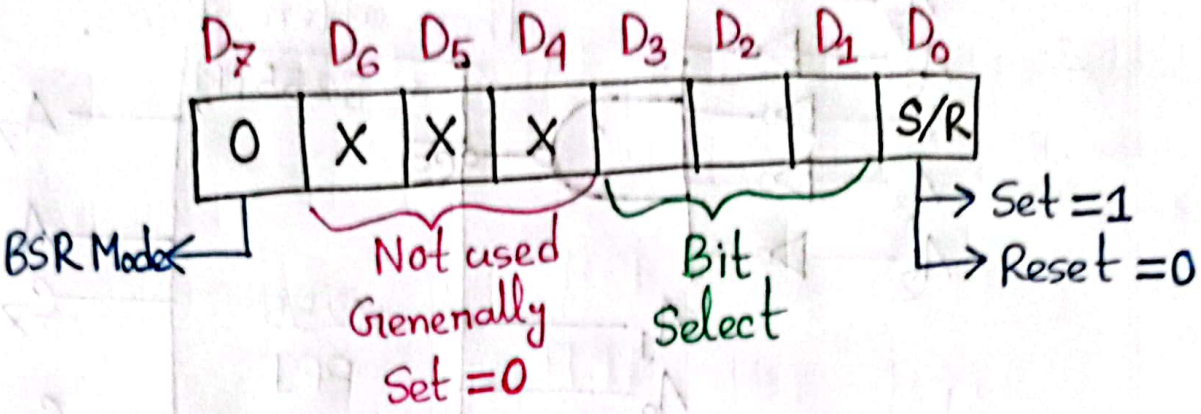
- The BSR mode is concerned only with the eight bits of port C. These 8-bits of port C can be set or reset by writing an appropriate control word in the control register.
- A control word with bit $D_7 = 0$ is recognized as a BSR control word.
- It does not change any previously transmitted control word with bit $D_7 = 1$.
- Thus, the I/O operations of ports A and B are not affected by a BSR control word.

In the BSR word, individual bits of port C can be used for application such as a ON/OFF switch.

Sweet Bird

Subject:.....
Date:.....Time:.....

BSR Control Word:



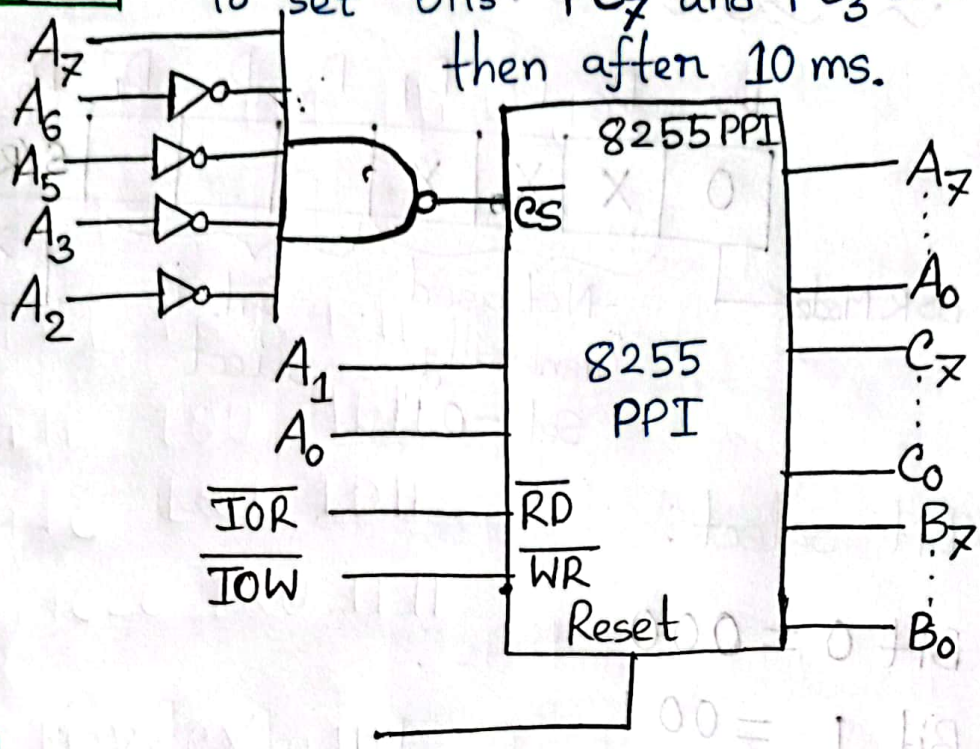
#Bit Select:

Bit 0 = 000
Bit 1 = 001
Bit 2 = 010
Bit 3 = 011
Bit 4 = 100
Bit 5 = 101
Bit 6 = 110
Bit 7 = 111

Sweet Bird

Subject:.....
Date:.....Time:.....

Problem: Write BSR control word subroute to set bits PC_7 and PC_3 and reset then after 10 ms.



Solution:

Port Addresses:

	A_{15} ---- A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	
PortA →	0 ---- 0	1	0	0	0	0	0	0	0	= 0080 H
	0 ---- 0	1	0	0	0	0	0	0	1	= 0081 H
	0 ---- 0	1	0	0	0	0	0	1	0	= 0082 H
	0 ---- 0	1	0	0	0	0	0	1	1	= 0083 H

BSR Control Words:

	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	
To set bit PC_7 =	0	0	0	0	1	1	1	1	= 0FH
To reset bit PC_7 =	0	0	0	0	1	1	1	0	= 0EH
To set bit PC_3 =	0	0	0	0	0	1	1	1	= 07H
To reset bit PC_3 =	0	0	0	0	0	1	1	0	= 06H.

Sweet Bird

Subject:.....
Date:.....Time:.....

Program:

```
MOV AL, 0FH
OUT 0083H, AL
MOV AL, 07H
OUT 0083H, AL
```

To set the ports: PC_7 and PC_3

$PC_3 = PC_7 = 1$

```
CALL DELAY
```

→ Delay for 10 ms

```
MOV AL, 0EH
OUT 0083H, AL
MOV AL, 06H
OUT 0083H, AL
```

To reset the ports: PC_7 and PC_3

$PC_3 = PC_7 = 0$

Note:

(1.) The BSR control word is always in control register not in Port C.

(2.) A BSR control word affects only one bit in port C.

(3.) The BSR control word does not affect the I/O mode.

The 8254 Programmable Interval Timer (8254 PIT)

Subject:.....
Date:.....Time:.....

#Why 8254 is used with 8086 microprocesson?

→ It is not possible to generate accurate time delays using Delay Routines in 8086.

The 8254 PIT provides:

- Accurate time delays
- Minimizes load on MP
- Real time clock
- Even Counter
- Digital one shot
- Square wave generator
- Complex Wave generator

Differences between 8254 and 8253:

8254	8253
1.) Can operate with higher clock frequency range. (DC to 8 MHz and 10MHz)	1.) Can be operated with comparatively lower frequency range (DC to 2MHz).
2.) Includes a Status Read-Back command that can latch the count and status of the counter.	2.) No Status-Read Back Command.
3.) Provides accurate result.	3.) Doesn't provide accurate result.
4.) It is a new version PIT.	4.) Old version PIT.

Sweet Bird

#Question: What is the advantages of high frequency range?

→ Increase information range, bandwidth range.

→ Increase counting range.

Block Diagram of 8254 PIT

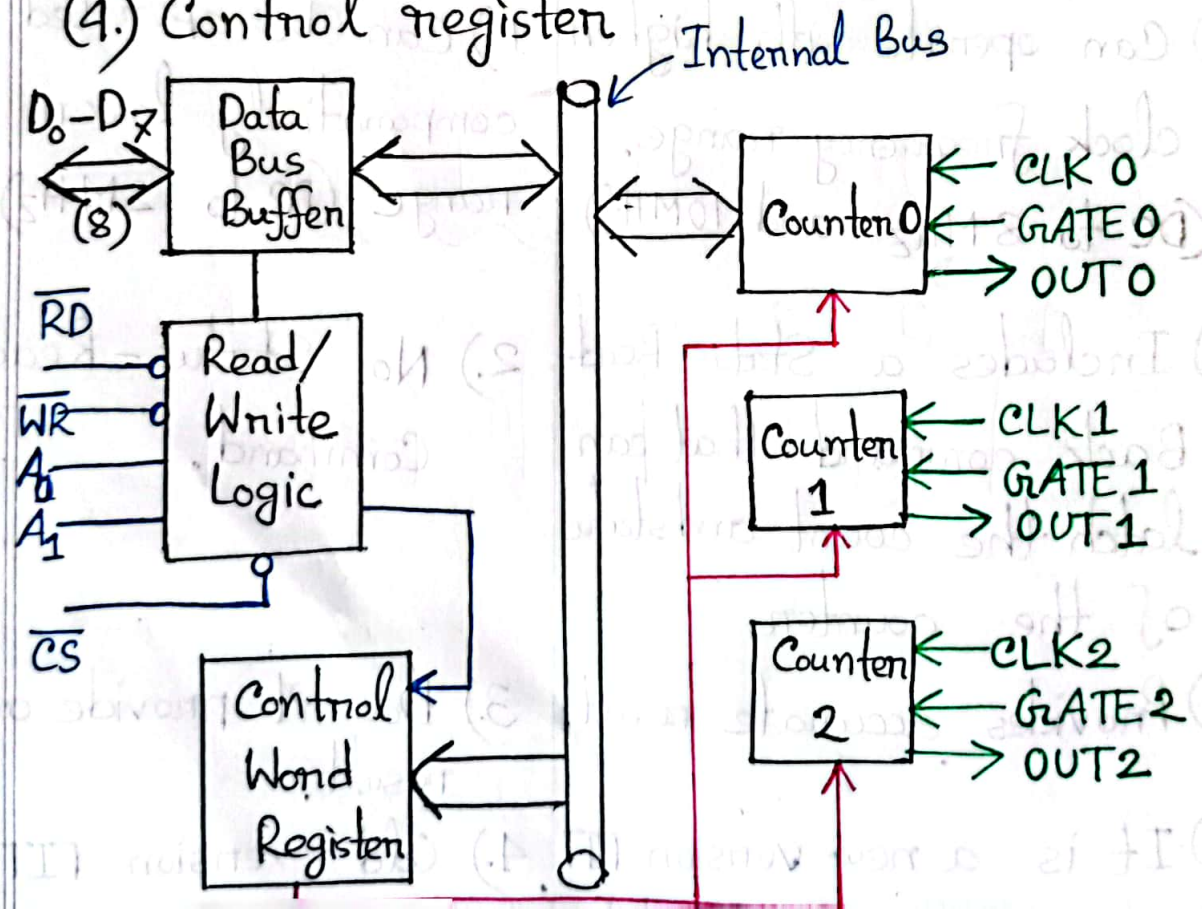
#Includes the following:

(1.) Three counters (0,1 and 2) → CLK, Gate } input; OUT } output

(2.) Data bus buffer.

(3.) Read/Write Control Logic

(4.) Control register



Subject:.....
Date:.....Time:.....

Control Word Format

SC1 SC0

0	0	Counter 0
0	1	Counter 1
1	0	Counter 2
1	1	Read-Back Command

SC - Select Counter

M - Mode:

M2	M1	M0	Mode
0	0	0	Mode 0
0	0	1	Mode 1
X	1	0	Mode 2
X	1	1	Mode 3
1	0	0	Mode 4
1	0	1	Mode 5

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
SC1	SC0	RW1	RW0	M2	M1	M0	BCD/Binary

BCD Mode:

0	Binary Counter -16 bits
1	BCD - 4 decades

RW on Read - Write:

RW1	RW0	Mode
0	0	Counter Latch Command (see Read Operation)
0	1	Read/Write LSB byte only
1	0	Read/Write MSB byte only
1	1	Read/Write LSB byte first then MSB byte

⇒ 16-bit can not be processed at a time, 8-bit + 8-bit

Sweet Bird

Subject:.....
Date:.....Time:.....

Modes of Operation

Mode-0: Interrupt on teminal count.

Mode-1: Programmable monoshot (Monstable)

Mode-2: Rate Generator.

Mode-03: Square wave generator.

Mode-4: Software Triggered Strobe. (Call-function)

Mode-05: Handware-Triggered Strobe.

- This is faster than Software Triggered
- Used to add additional devices

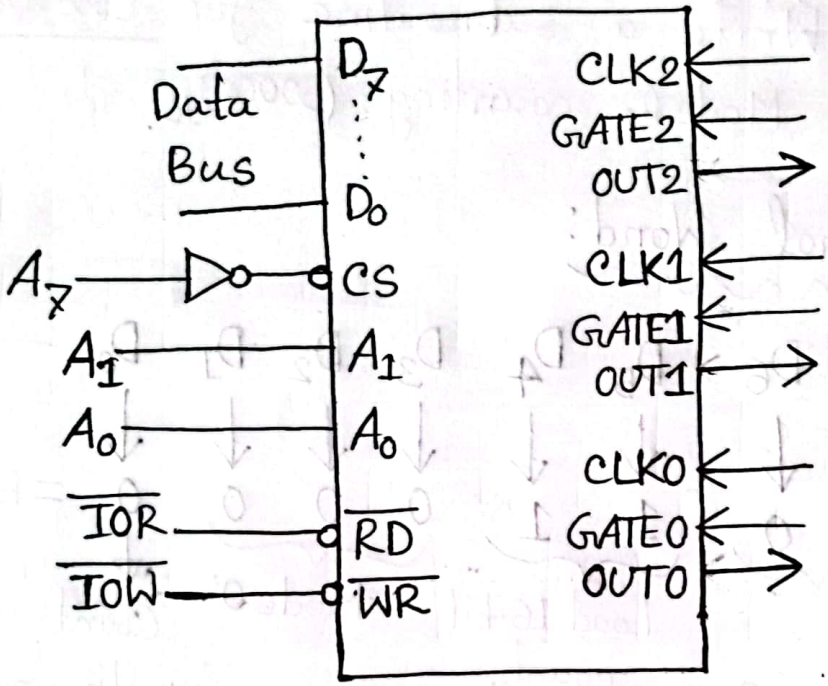
Control Word Format

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
SC_1	SC0	RW1	RW0	M_2	M_1	M_0	BCD

Subject:.....

Date:.....Time:.....

Question: The 8254 as a Counter.



(1) Identify the port address of the control register and counter 2.

Solution: The port addresses are:

A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	
↓	↓	↓	↓	↓	↓	↓	↓	
1	0	0	0	0	0	0	0	= 80H → Counter 0
1	0	0	0	0	0	0	1	= 81H → Counter 1
1	0	0	0	0	0	1	0	= 82H → Counter 2
1	0	0	0	0	0	1	1	= 83H → Control Register

Control Register = 83H

Counter 2 = 82H.

Sweet Bird

(2.) Write a program for counter-2 in Mode 0, count for $(50000)_{10}$.
Or, Write a subroutine for counter-2 in Mode 0, counting $(50000)_{10}$.

Control Word:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	
↓	↓	↓	↓	↓	↓	↓	↓	
1	0	1	1	0	0	0	0	= B0H.

Select Counter-2 | Load 16-bit count, $(50000)_{10}$ first load LSB then MSB | Mode 0 | Count in binary

The value $= (50000)_{10} = 1100\ 0011\ 0101\ 0000$
(MSB LSB)
$= C350H.$

Program:

```
MOV AL, B0H   } CW → CR
OUT 83H, AL   }

MOV AL, 50H   } LSB Out → Counter 2
OUT 82H, AL   }

MOV AL, C3H   } MSB Out → Counter 2
OUT 82H, AL   }
```

Subject:.....
Date:.....Time:.....

Example : 15.4 1 kHz Square Wave generates at Counter-1 from the previous figure.

Write a program to generate 1 kHz square wave from Counter-1.

→ The Control Word :

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
↓	↓	↓	↓	↓	↓	↓	↓
0	1	1	1	0	1	1	0
Counter-1 Select		Load 16-bit count		Mode-3 (Square-wave generator)			Binary Count Bitwize

= 76 H.

The control addresses :

Counter-1 = 81 H.

Control Register = 83 H.

Note that: If there is no count value given, it should be calculated manually.

→ We know that -

The clock freq. of 8254 is between 0 - 8 MHz.

Sweet Bird

Date:.....Time:

Let, the clock frequency is = 2MHz.

$$\therefore T = \frac{1}{f} = \frac{1}{2\times10^6} = 0.5\,\mu s.$$

The square wave's frequency,

$$f = 1\,kHz.$$

$$T_{square} = \frac{1}{1\times10^3} = 1\,ms.$$

$$\therefore \text{Required no. of counts} = \frac{T_{square}}{T}$$

$$= \frac{1\times10^{-3}}{0.5\times10^{-6}}$$

$$= 2000.$$

$$= 07D0\,H$$

MSB LSB

Program:

```
MOV AL, 76H   }
OUT 83H, AL   }  CR → CW

MOV AL, D0H   }
OUT 81H, AL   }  LSB OUT

MOV AL, 07H   }
OUT 81 H, AL  }  MSB OUT
```

Sweet Bird