Design an Interfacing 8254 with 8085 Processor

A simple schematic for interfacing the 8254 with an 8085 processor is shown in diagram.1. The 8254 can be either memory-mapped or IO-mapped in the system. In the schematic shown in diagram.1. The 8254 is IO-mapped in the system. The chip select signals for IO-mapped devices are generated by using a 3-to-8 decoder. The address lines A_4, A_5 and A_6 are decoded to generate eight chip select signals (IOCS-0 to IOCS-7) and in this, the chip select IOCS-5 is used to select the 8254. The address line A_7 and the control signal IO/$\overline{M}$ are used to enable the decoder.

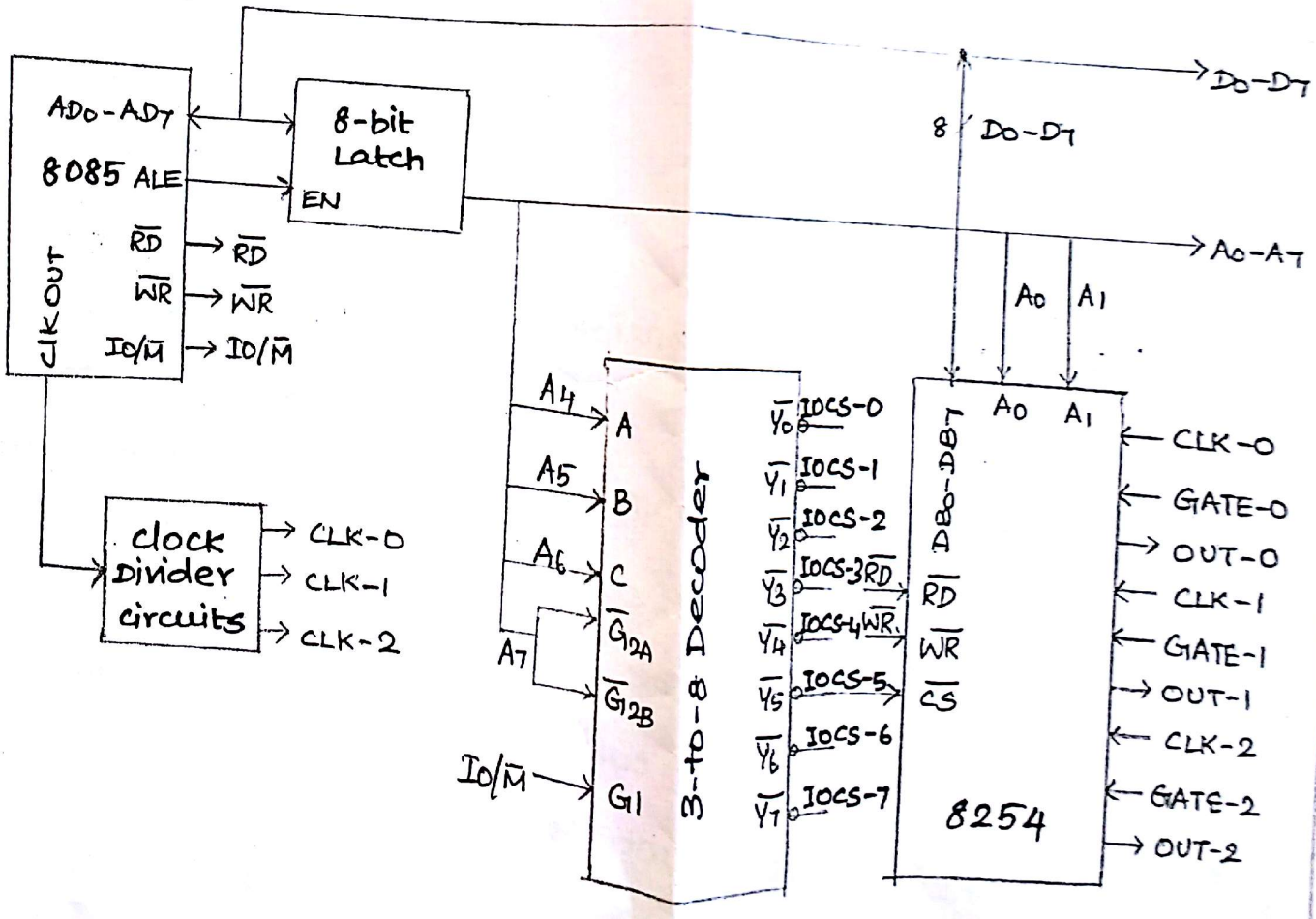

Diagram.1

The address lines A_0 and A_1 of the 8085 are connected to A_0 and A_1 of the 8254 to provide the internal address. The IO addresses alloted to the internal devices of 8254 are listed in below table. The data lines D_0-D_7, $\overline{RD}$ and $\overline{WR}$ signals of the 8254 are connected to the D_0-D_7, $\overline{RD}$ and $\overline{WR}$ of the processor respectively to achieve parallel data transfer.

Table: IO Addresses of 8254 interfaced to 8085 As shown in diagram.1

Internal device	BINARY ADDRESS								Hexa address
	Decoder Input and enable				Input to address pins of 8254				
	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	
counter-0	0	1	0	1	X	X	0	0	50
counter-1	0	1	0	1	X	X	0	1	51
counter-2	0	1	0	1	X	X	1	0	52
control Register	0	1	0	1	X	X	1	1	53

Note: Don't care "x" is considered as zero.

The clock signals required for the counters can be obtained either from the processor clock output or from an external clock source. The clock signal from a 8085 can also be divided to lower values by using clock divider circuits and then applied to clock input of counters.

A switch is connected to pin P2.7. write a program to monitor the status of SW and perform the following:

a) If SW=0, the stepper motor moves clockwise

b) If SW=1, the stepper motor moves counter clockwise.

PROGRAM:

```
    ORG   0H          ; starting address
MAIN:
    SETB  P2.7        ; make an input
    MOV   A, #66H     ; starting phase value
    MOV   P1, A       ; send value to port
TURN:
    JNB   P2.7, CW    ; check switch result
    RR    A           ; rotate right
    A CALL DELAY      ; call delay
    MOV   P1, A       ; write value to port
    SJMP  TURN        ; repeat
CW: RL    A           ; rotate left
    A CALL DELAY      ; call delay
    MOV   P1, A       ; Write value to port
    SJMP  TURN        ; repeat
DELAY:
    MOV   R2, #100
H1: MOV   R3, #255
H2: DJNZ  R3, H2
    DJNZ  R2, H1
    RET
    END
```

Algorithm:-

1. Start
2. Make an input P2.7
3. Start the phase value and send the value to port
4. Check the switch result and rotate to right
5. call a delay and write the value to port and repeat
6. rotate left again and call a delay.
7. write the values to port and repeat the process