PROJECT REPORT ON

A MICROPROCESSOR-BASED IMAGE SCANNER

(2020-21)

Praveen Kumar Singh	2019AAPS0221G
Akshat	2019A8PS0492G
Saksham Bansal	2019A7PS0056G
Aakarsh Goyal	2019A3PS0096G
Prakhar Jain	2019A3PS0370G
Akshat Agrawal	2019A3PS0292G

PROBLEM STATEMENT (NUMBER 5)

Design a microprocessor-based scanner which will scan a black and white image and store it as binary data.

The scanner has two stepper motors for motion along two orthogonal coordinates. The rotational motion is converted into transnational motion through a lead-screw mechanism.

Five paired LED photodiodes intended for B&W image scanning are placed 0.1 centimetre apart. The maximum size scannable is 10cm X 10cm.

The photodiode output is analog signal (between 0 to 5 Volts) which is to be digitized.

Image information is stored sequentially in the RAM.

The user presses a switch labelled **Start Scan** when he wants the scanning process to be completed. Once scanning is completed an LED labelled **Scan Complete** will glow.

PROBLEM DESCRIPTION

The Microprocessor based Black and White image scanner consists of the INTEL 8086 MPU as a central processing unit.s

Two stepper motors are used to move the scanner in two orthogonal directions.

The photodiode absorbs the light with various intensities forming an analog signal (0-5V) . This analog signal is converted into a digital signal through an analog to digital converter 0808.

The 5 paired photodiodes are interfaced to the system through the ADC 0808.

The digital output of the black and white image is stored in the RAM which is interfaced to the processor.

ASSUMPTIONS

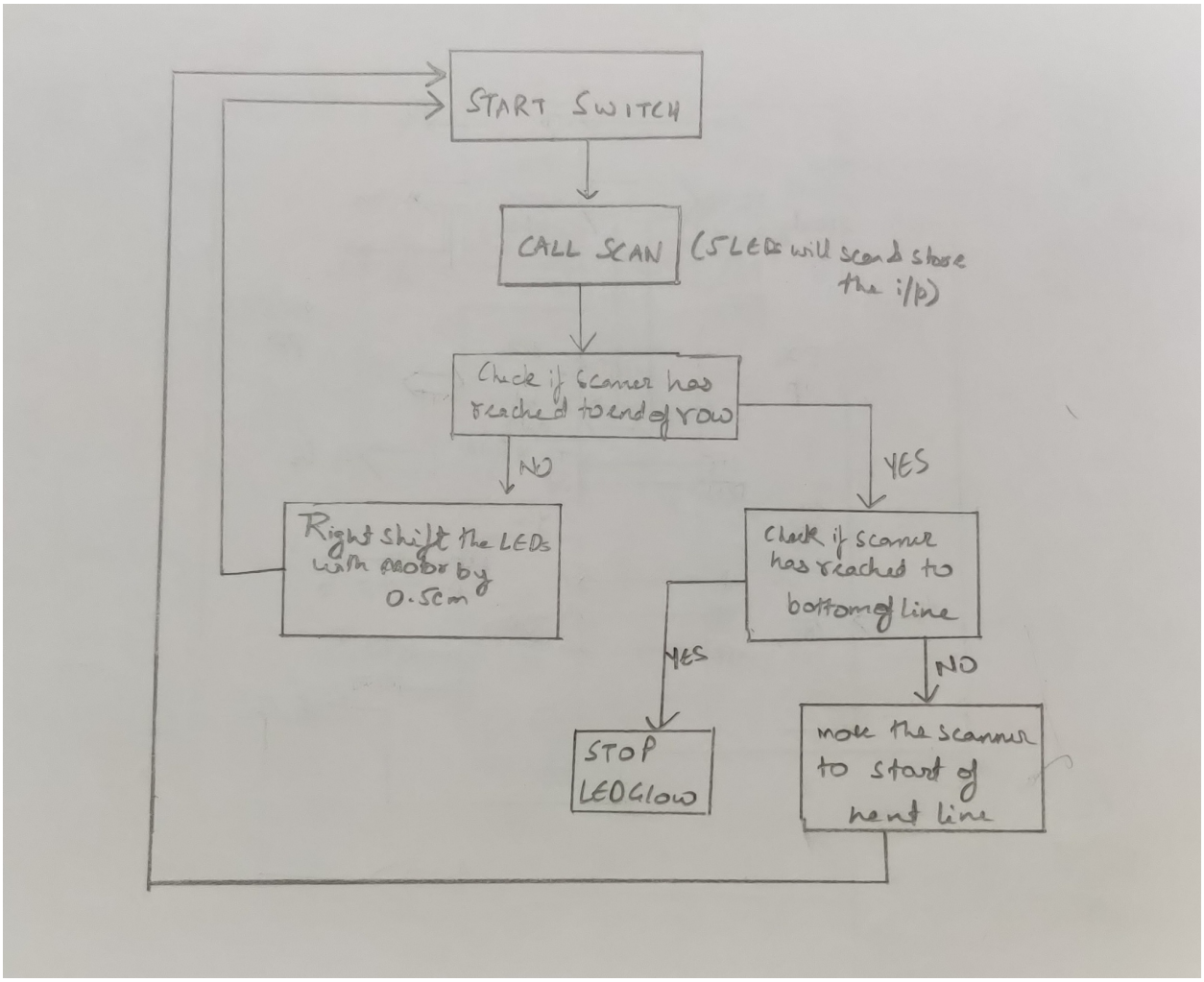
- The image to be scanned has been assumed to be smooth and perfectly reflecting.
- There is assumed to be no power failure while the system is operating.
- For one full rotation, the stepper motor covers 0.5 cm
- Size of the pixel is taken as 0.1cm.
- The LED is assumed to be scanning 1 pixel at a time.
- Scanning bigger images is possible but we have to modify our project. So we are restricting the size of 10x10 cm.
- We will get a range of analog signals from 0 to 5 volt.
- Delay time of 2 milliseconds is taken for the stepper motor to rotate into its position.
- The starting position is from the top-left corner of the page. Motion of the photodiodes would be from left to right and top to bottom of a page in the read operation.
- The system waits for the signal by the user into the reset pin to start the scanning process.

ICs USED

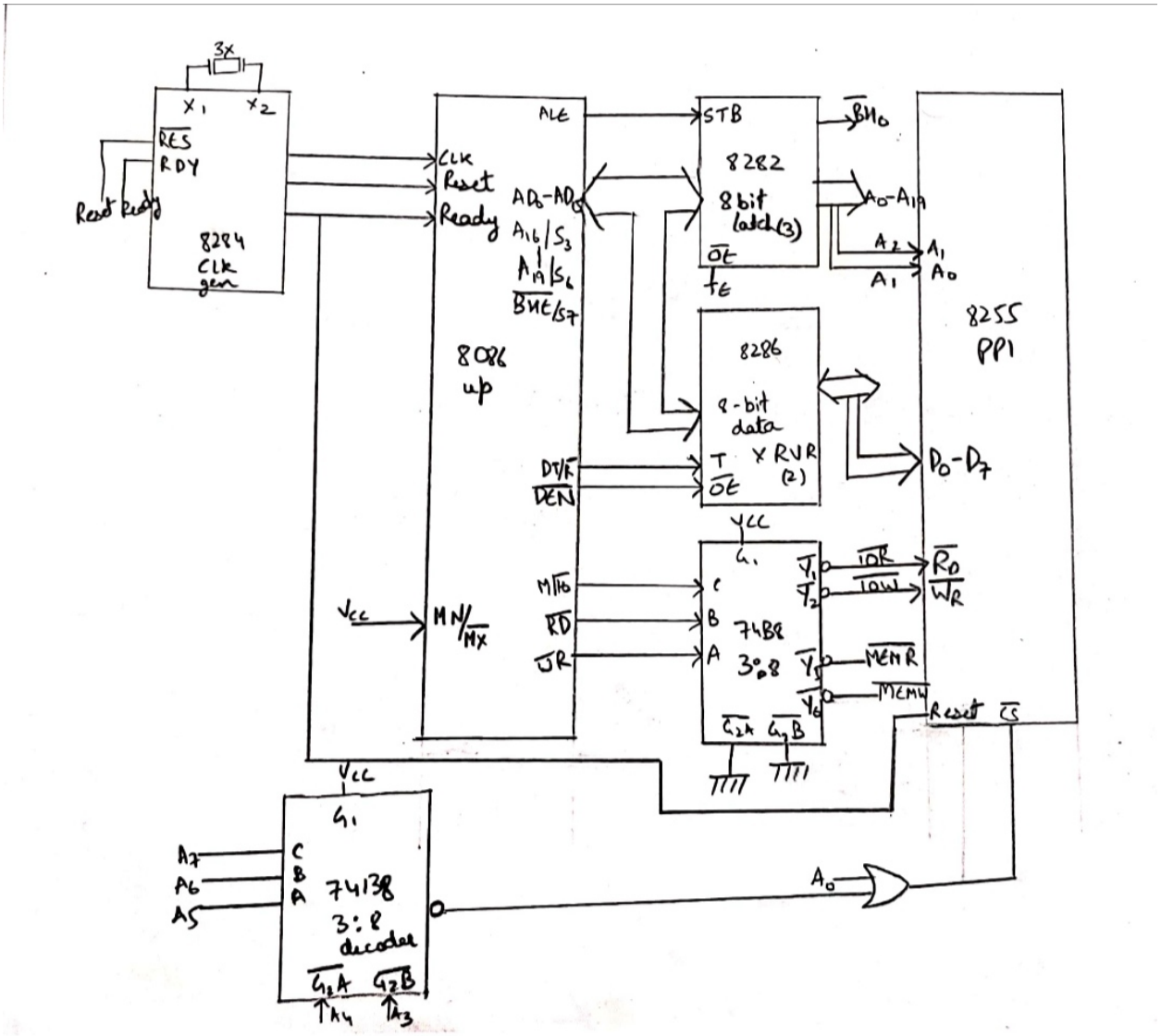
COMPONENT NAME	NUMBER OF COMPONENTS USED	PURPOSE
INTEL 8086	1	central processing unit
INTEL 8255A	2	To interface a microprocessor with input ports and out ports.
ULN2003A	1	Used to drive the motor
8286	1	To separate address lines and data lines
2732	2	4 KB ROM
2064	2	8 KB RAM
74LS138	1	Decoder
ADC 0808	1	To convert analog signal to digital signal
74HC32	8	OR Gate
Stepper Motor	1	Used for movement in orthogonal directions
7488	1	3:8 Decoder

74HC4078	1	NOR Gate
74HC04	4	Inverter

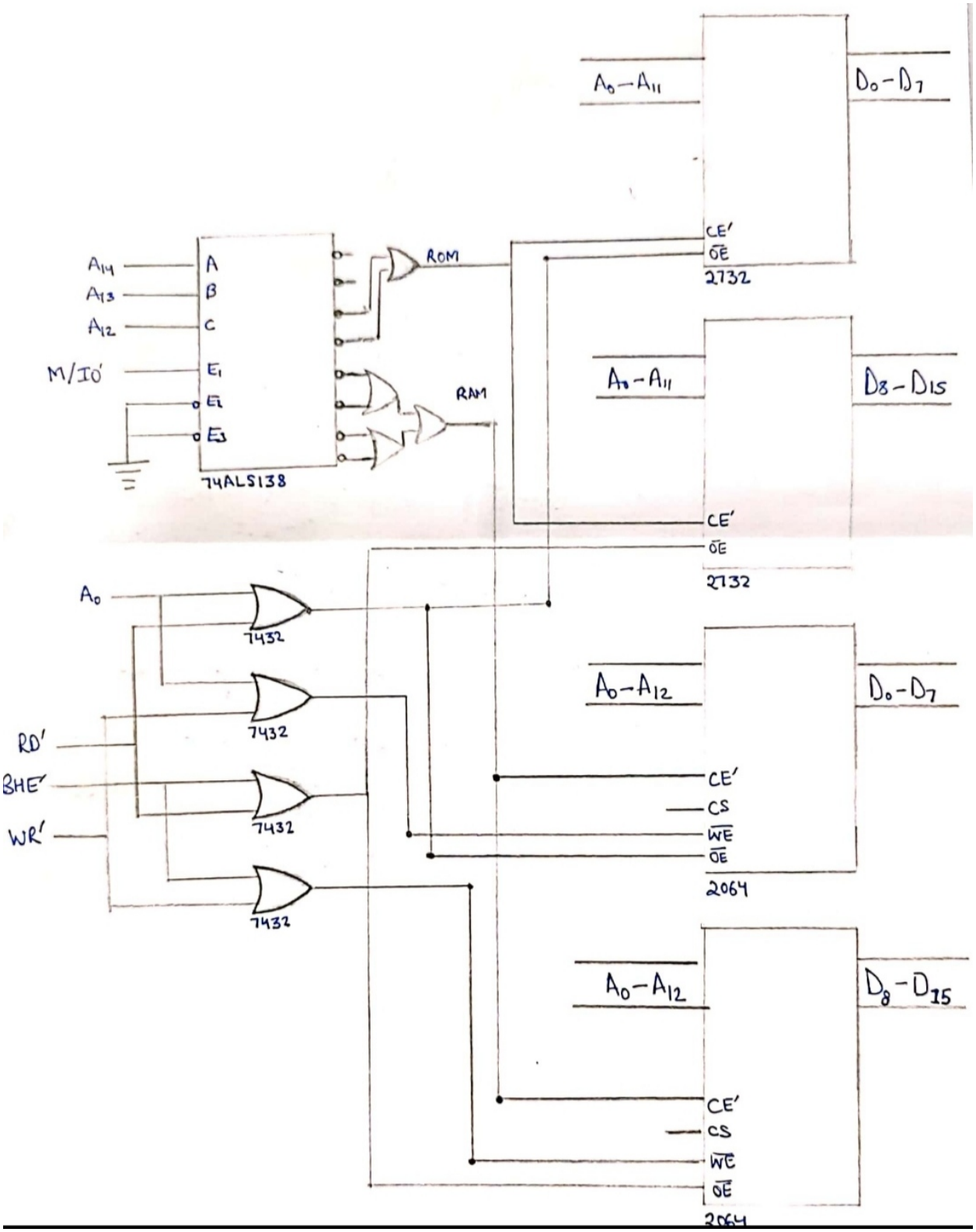
Block Diagram



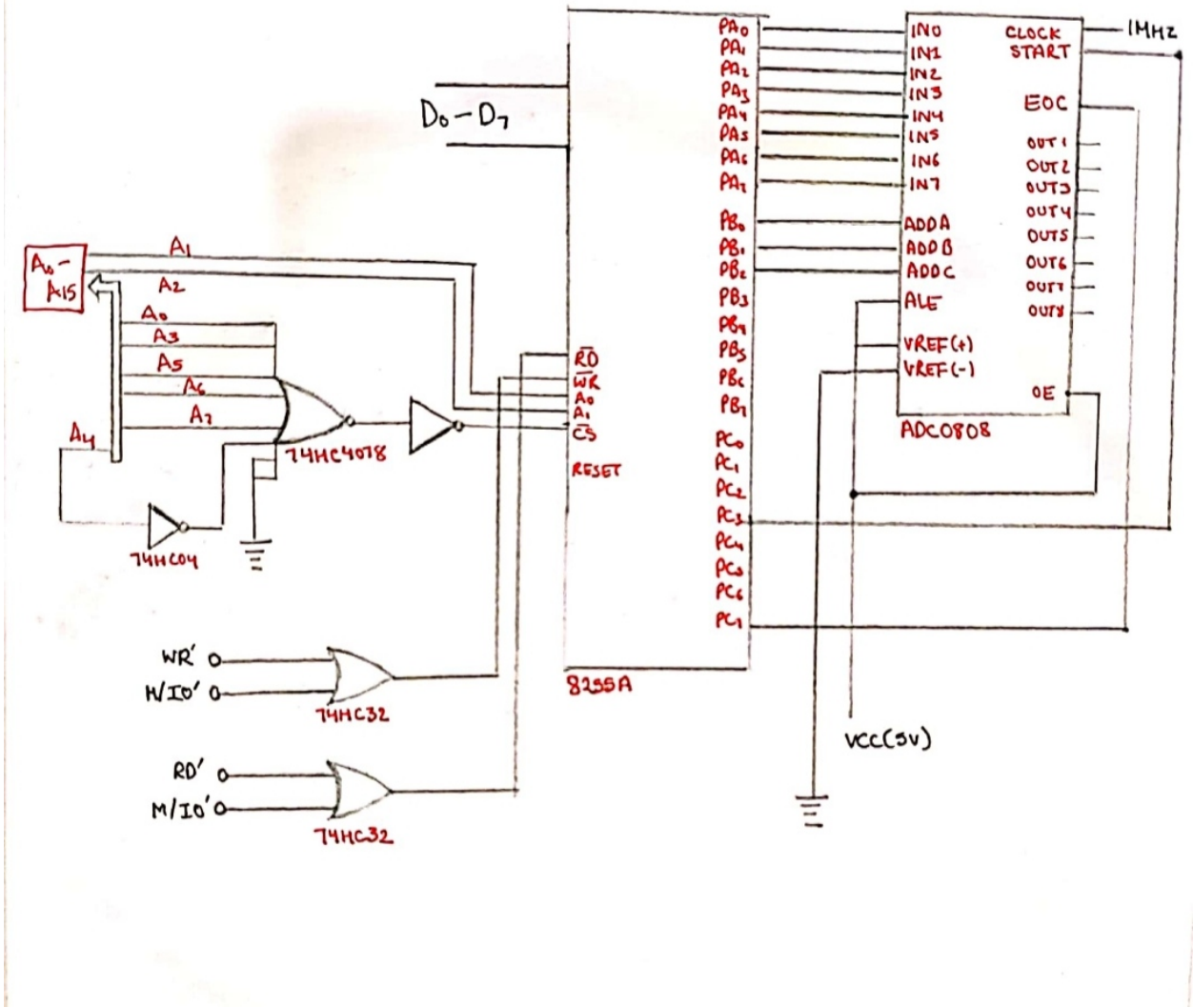
Interfacing 8086 with 8255

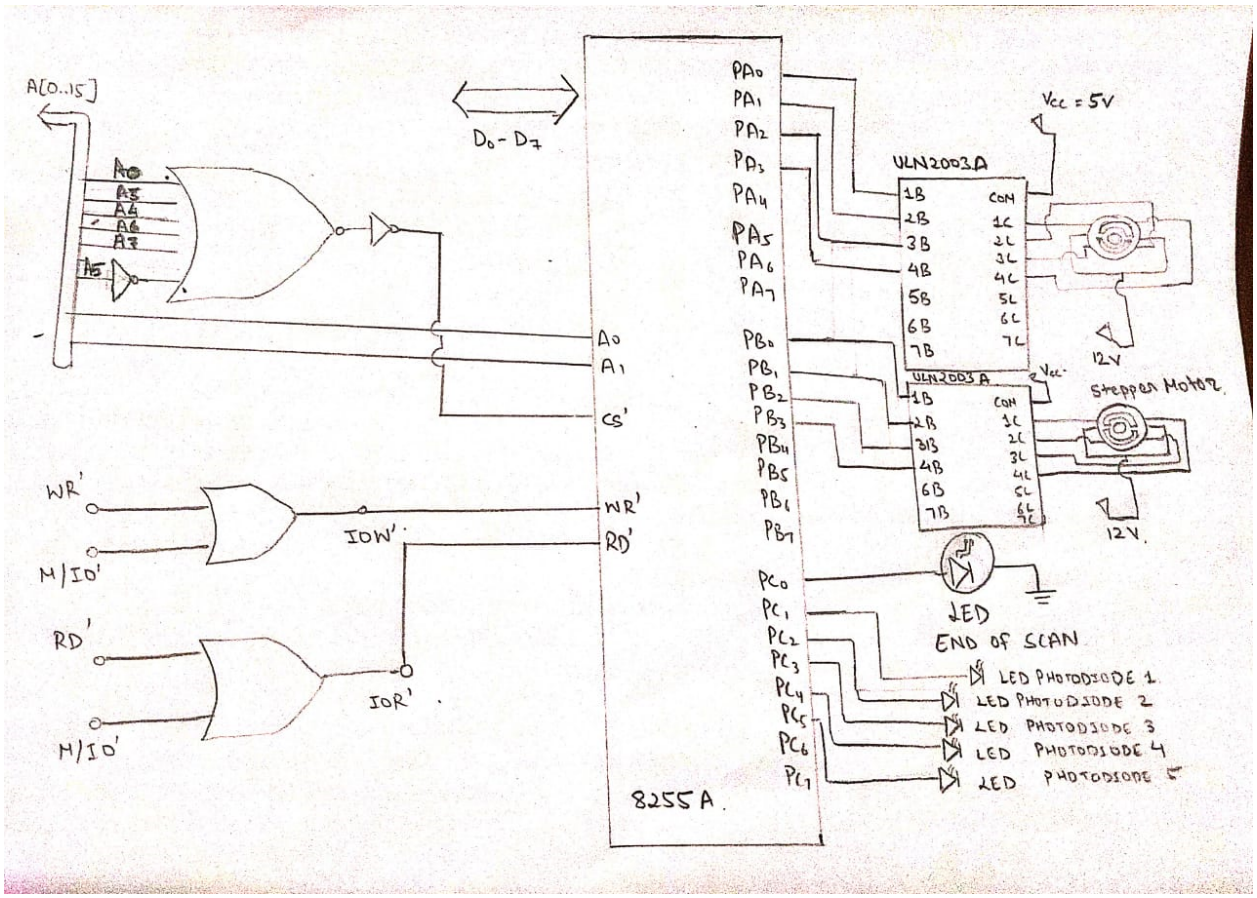


Interfacing memory with 8255

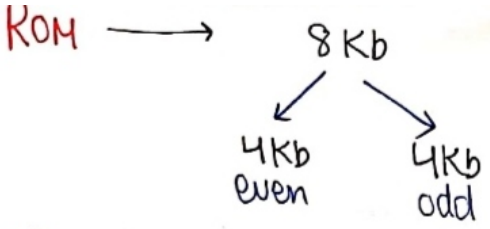


Interfacing 8255 with 0808




A[0..15]
A0
A5
A4
A6
A7
A5
D0 - D7
A0
A1
CS'
WR'
M/IO'
IOW'
WR'
RD'
RD'
M/IO'
IOR'
8255A
PA0
PA1
PA2
PA3
PA4
PA5
PA6
PA7
PB0
PB1
PB2
PB3
PB4
PB5
PB6
PB7
PC0
PC1
PC2
PC3
PC4
PC5
PC6
PC7
ULN2003A
1B
2B
3B
4B
5B
6B
7B
COM
1C
2C
3C
4C
5C
6C
7C
Vcc = 5V
12V
Stepper Motor
ULN2003A
Vcc
12V.
LED
END OF SCAN
LED PHOTODIODE 1
LED PHOTODIODE 2
LED PHOTODIODE 3
LED PHOTODIODE 4
LED PHOTODIODE 5

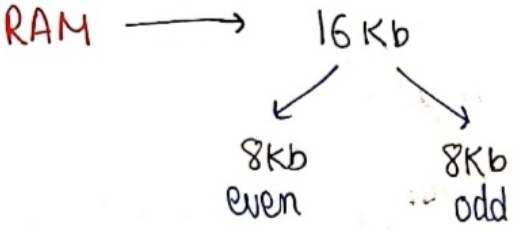
Memory Mapping



ROM 1 (e) :- 2000_H ---------- $3FFE_H$

ROM 1 (o) :- 2001_H ---------- $3FFF_H$

Total :- 2000_H ----- $3FFF_H$



RAM 1 (e) :- 4000_H ---------- $7FFE_H$

RAM 1 (o) :- 4001_H ---------- $7FFF_H$

Logic

A_{15}	A_{14}	A_{13}	A_{12}	
0	0	1	0	ROM 1
0	0	1	1	
0	1	0	0	RAM 1
0	1	0	1	
0	1	1	0	
0	1	1	1	

ALP

```
.model tiny
.data
        ; each pixel = 0.1 cm

        ; for ADC
        porta1 equ 10h          ; input -  data lines between ADC and 8255
        portb1 equ 12h          ; output - interfacing between adc and 8255 - to select input pin of
ADC(AD0, AD1, AD2)
        portc1 equ 14h          ; upper-i/p, lower-o/p - interfacing between ADC and 8255 - control
logic
        creg1 equ 16h

        ;for motor
        porta2 equ 20h          ; responsible for horizontal movement, pitch of lead screw =
2.5cm, rotor teeth = 200(200 rotations for 1 complete revolution), BX
        portb2 equ 22h          ; responsible for vertical movement, pitch of lead screw = 0.5cm,
rotor teeth = 200(200 rotations for 1 complete revolution), DX
        portc2 equ 24h
        creg2 equ 26h

        horcnt equ 20 ; horizontal count, since horizontally we need to move (2*x) times, where x
= 10cm
        vercnt equ 100 ; vertical count, since vertically we need to move (10*x) times, where x =
10cm


        store db 10000 DUP(0)       ; 100*100 pixels and every pixel = 1Byte
        stack dw 10 DUP(0)
        top_stack label word

.code
.startup
                lea sp, top_stack
                mov AX, 2000h       ;
                mov ds, ax          ; initializing DS

                lea di, store   ;
                mov bx, 00          ; horizontal counter
                mov dx, 00          ; vertical counter
```

```
X1:             CALL SCAN

                CMP BX, horcnt      ; do the scanning again after shifting if not reached at the
end of the row.
                JNE shift           ; (call scanning function in shift)
                CMP DX, vercnt      ; check if it reached to last row, if not then jump to nextline
                JNE nextl

                JMP stop            ; to turn the stop led on.


                ; to shift the LEDs by 0.5cm in a row
shift:  MOV al, 80h         ;
                out creg2, al   ;
                mov al, 88h         ; we want o/p = 10001000 at porta2
                mov cx, 40          ; to shift by 0.5cm(pitch/5) we need 200/5 revolutions
loop1:  out porta2, al  ;
                call delay1         ; delay1 function to wait for atleast 10ms = internal delay1
of motor
                ROR al, 01          ; clockwise direction is chosen to move shift right in a row
                loop loop1

                JMP X1                      ; jump to scan again after rotating the motor.


                ; to move leds to starting of next line
nextl:  MOV al, 80h
                out creg2, al
                mov al, 88h
                mov cx, 760     ; to move the LEDs from end of the row to the starting. no. of times
loop reqd = 40*19
loop2:  out porta2, al
                call delay1
                ROL al, 01          ; anticlockwise direction to shift left the leds
                loop loop2

                MOV al, 80h
                out creg2, al
                mov al, 88h
                mov cx, 40          ; to move down by 0.1cm(pitch/5) we need 200/5
revolutions
loop3:  out portb2, al
                call delay1
                ROR al, 01          ; clockwise direction is chose to move down in a column
```

```
                loop loop3

                inc dx                  ; increment vertical counter

                JMP X1


stop:   mov al, 80h             ; 10000000b - all port are output
                out creg2, al   ; initializing port2

                mov al, 01h             ; 00000001b - giving PC0 of port2 'stop' signal
                out creg2, al   ; output to PC0 given to turn on stop led



.exit



SCAN Proc Near

        mov al, 98h     ;10011000b since only data lines(porta1) and upper portc1 are input and
mode 0 -
        out creg1, al   ;initializing port1

        Scan1: mov al, 06h              ; 00000110b - PC3 = SOC low pulse
                        out creg1, al

                        mov al, 01h             ; 01 is the ADC port
                        out portb1, al  ; using B to select 1st input of ADC

                        ;we will be keeping ALE and OE at 5V all time.

                        ; giving SOC a pulse

                        mov al, 07h     ; 00000111b - PC3 = SOC high pulse
                        out creg1, al
                        call delay2                             ; use sufficient nop such that 8255
can read it..becz 86 runs at x5 speed than 8255
                        mov al, 06h     ; 00000110b - PC3 = SOC low pulse
                        out creg1, al

        check11:in al, portc1   ; checking input of port C to know EOC value
                        mov cl, 1
```

```
                rcl al, cl
                jc check11          ; this is to check if EOC is low, if not repeat

check12:in al, portc1
                mov cl,1
                rcl al, cl
                jnc check12         ; this is to check if EOC is high, if not then dont
jump

                in al, porta1   ;
                mov [di], al    ; moving the data into store loc.
                inc di              ;




Scan2: mov al, 06h          ; 00000110b - PC3 = SOC low pulse
                out creg1, al

                mov al, 02h         ; 02 is the ADC port
                out portb1, al  ; using B to select 1st input of ADC

                ;we will be keeping ALE and OE at 5V all time.


                ; giving SOC a pulse

                mov al, 07h     ; 00000111b - PC3 = SOC high pulse
                out creg1, al
                call delay2                     ; use sufficient nop such that 8255
can read it..becz 86 runs at very high speed than 8255
                mov al, 06h     ; 00000110b - PC3 = SOC low pulse
                out creg1, al

check21:in al, portc1   ; checking input of port C to know EOC value
                mov cl, 1
                rcl al, cl
                jc check21          ; this is to check if EOC is low, if not repeat

check22:in al, portc1
                mov cl,1
                rcl al, cl
```

```
                    jnc check22             ; this is to check if EOC is high, if not then dont
jump


                    in al, porta1   ;
                    mov [di], al    ; moving the data into store loc.
                    inc di


      Scan3: mov al, 06h            ; 00000110b - PC3 = SOC low pulse
                    out creg1, al

                    mov al, 03h             ; 03 is the ADC port
                    out portb1, al  ; using B to select 1st input of ADC

                    ;we will be keeping ALE and OE at 5V all time.


                        ; giving SOC a pulse

                    mov al, 07h     ; 00000111b - PC3 = SOC high pulse
                    out creg1, al
                    call delay2                             ; use sufficient nop such that 8255
can read it..becz 86 runs at very high speed than 8255
                    mov al, 06h     ; 00000110b - PC3 = SOC low pulse
                    out creg1, al

      check31:in al, portc1  ; checking input of port C to know EOC value
                    mov cl, 1
                    rcl al, cl
                    jc check31              ; this is to check if EOC is low, if not repeat

      check32:in al, portc1
                    mov cl,1
                    rcl al, cl
                    jnc check32             ; this is to check if EOC is high, if not then dont
jump


                    in al, porta1   ;
                    mov [di], al    ; moving the data into store loc.
                    inc di
```

```
Scan4: mov al, 06h              ; 00000110b - PC3 = SOC low pulse
       out creg1, al

       mov al, 04h              ; 04 is the ADC port
       out portb1, al  ; using B to select 1st input of ADC

       ;we will be keeping ALE and OE at 5V all time.


       ; giving SOC a pulse

       mov al, 07h     ; 00000111b - PC3 = SOC high pulse
       out creg1, al
       call delay2                              ; use sufficient nop such that 8255
can read it..becz 86 runs at very high speed than 8255
       mov al, 06h     ; 00000110b - PC3 = SOC low pulse
       out creg1, al

check41:in al, portc1   ; checking input of port C to know EOC value
       mov cl, 1
       rcl al, cl
       jc check41              ; this is to check if EOC is low, if not repeat

check42:in al, portc1
       mov cl,1
       rcl al, cl
       jnc check42             ; this is to check if EOC is high, if not then dont
jump


       in al, porta1     ;
       mov [di], al      ; moving the data into store loc.
       inc di


Scan5: mov al, 06h              ; 00000110b - PC3 = SOC low pulse
       out creg1, al

       mov al, 05h              ; 05 is the ADC port
       out portb1, al  ; using B to select 1st input of ADC

       ;we will be keeping ALE and OE at 5V all time.
```

```
                        ; giving SOC a pulse

                        mov al, 07h     ; 00000111b - PC3 = SOC high pulse
                        out creg1, al
                        call delay2                             ; use sufficient nop such that 8255
can read it..becz 86 runs at very high speed than 8255
                        mov al, 06h     ; 00000110b - PC3 = SOC low pulse
                        out creg1, al

        check51:in al, portc1   ; checking input of port C to know EOC value
                        mov cl, 1
                        rcl al, cl
                        jc check51              ; this is to check if EOC is low, if not repeat

        check52:in al, portc1
                        mov cl,1
                        rcl al, cl
                        jnc check52             ; this is to check if EOC is high, if not then dont
jump


                        in al, porta1   ;
                        mov [di], al    ; moving the data into store loc.
                        inc di

        INC BX          ; increasing horizontal counter

        RET
SCAN ENDP

delay1 proc near
        push cx
                mov cx, 3000
loop4:  nop
                nop
                nop
                nop
                nop
                loop loop4
        pop cx
        ret
delay1 endp

delay2 proc near
```

```
                nop
                nop
                nop
                nop
        ret
delay2 endp

end
```