



**Microprocessor and Assembly  
Language Course Project**

**Elevator Design**

**Instructor: Dr. Ghazvini**

**Group Members:**

**Mehdi Borhani Nejad**

**Babak Fathi**

**Esmail Bagheri**

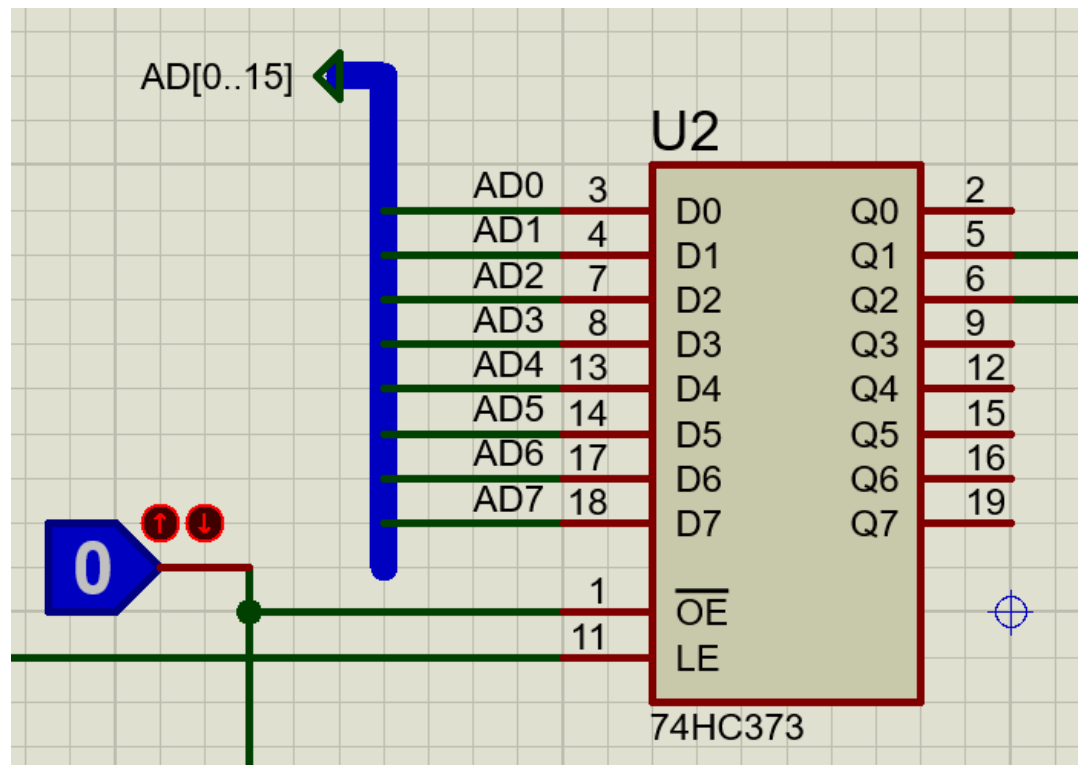
**We want to design an elevator system for a 6-story building. This system should include an emergency button. When pressed, the elevator should stop immediately at its current location and remain stationary for a few seconds. The elevator should stay on the floor where it stopped, and the emergency light should turn on.**

- **8086**
- **7seg**
- **74LS138**
- **8255A**
- **L298**
- **74HC373**



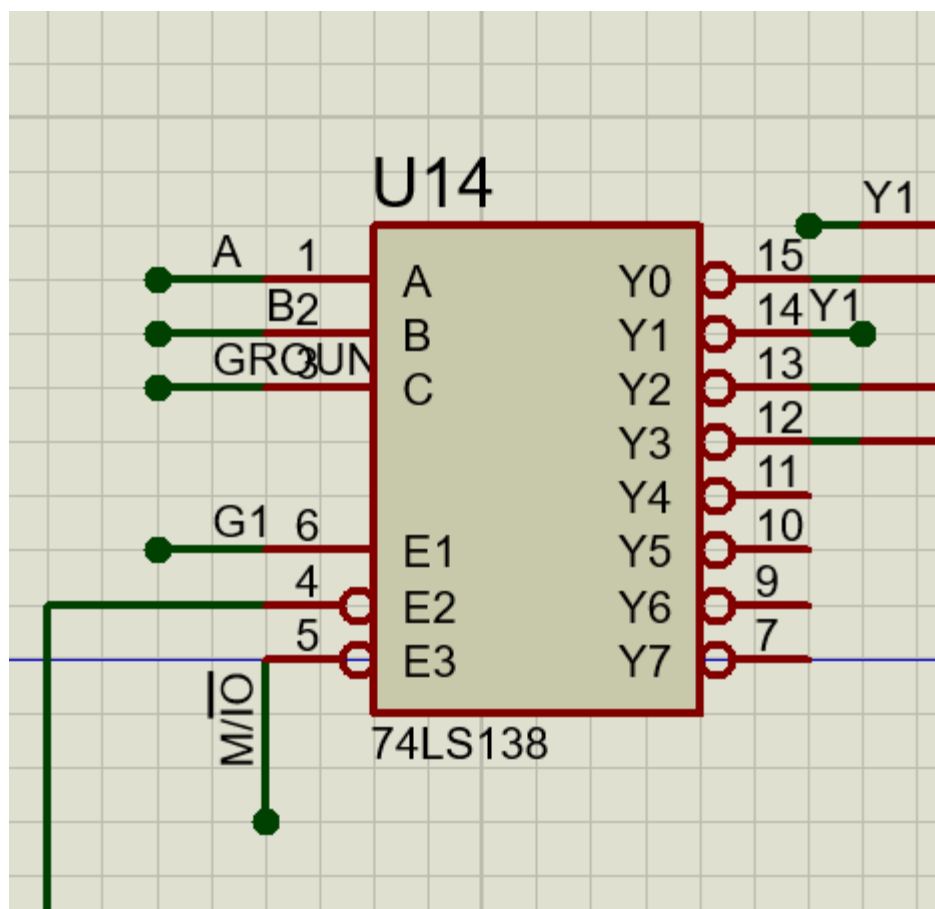
### 74HC373 (LATCH):

This latch has two pins: output enable and latch enable. The output enable pin is always connected to ground due to being active low, and the latch enable is activated when an address is placed on our databus.



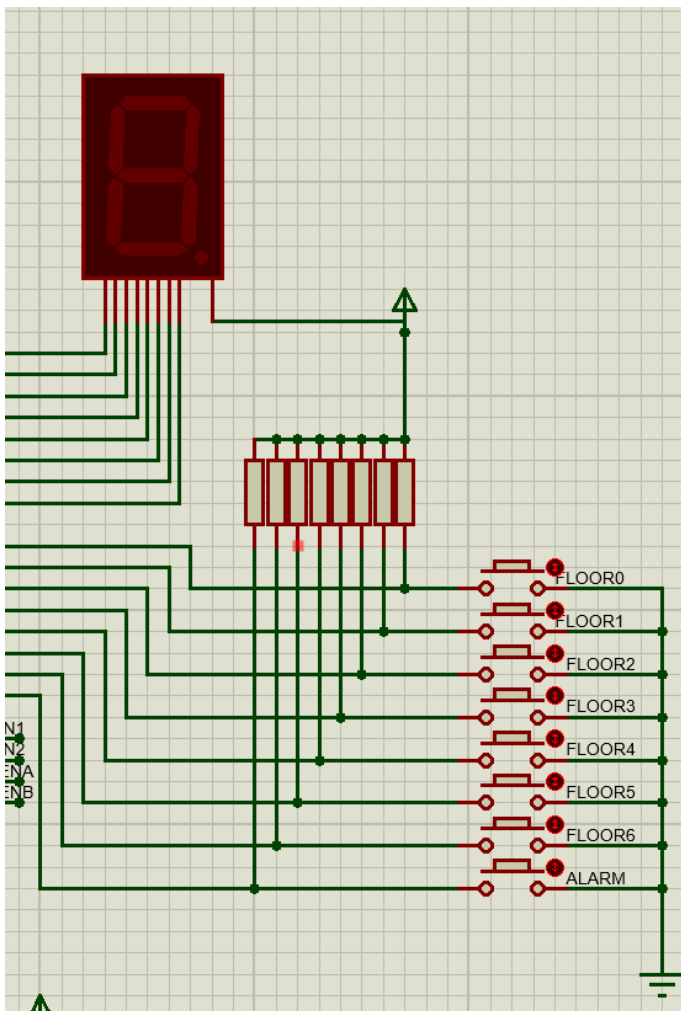
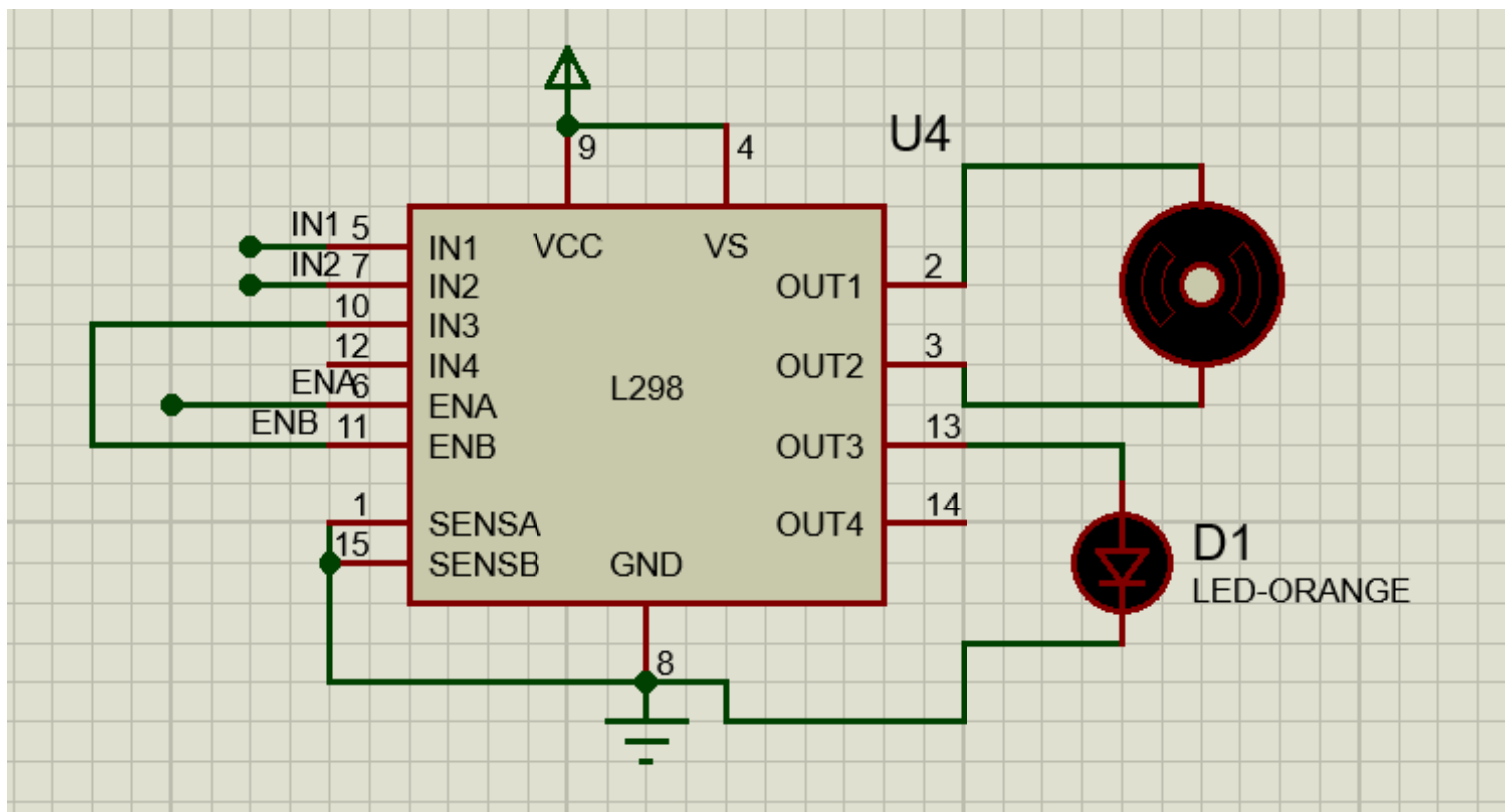
### 74LS138 (DECODER):

In this decoder, lines (A, B, C) are our selection lines that determine which I/O should start operating. Additionally, there are three enable lines in this decoder, with E2 being the most important one because it causes the decoder to start operating only through specific addresses.



### L298:

This component is responsible for activating our motor when going up or down and activating the warning buzzer when the elevator's alarm button is pressed.



### 7seg:

This component is in the form of a common cathode, which turns on the corresponding segment when one of its lines becomes zero.

### BUTTON:

Designed as pull-up, in this design when the button is pressed, its output value changes from 1 to 0.

**8255:**

The pin configuration of 8255 is shown in the figure below. In this chip, lines D0 to D7 are responsible for determining the state of the three I/O ports (A, B, C).

**D7 (most significant bit):** To specify that our system works as a simple input and output, we set it to 1.

**D5\_D6:** Determining the operating mode of A from three existing modes, which we must consider as a simple input-output. Therefore, both of these are set to 0.

**D4:** Determines the use of A as input (1) or output (0).

**D3\_D0:** PORTC can have both input and output roles. We divide its pins into two sections of low-value bits (PC0-3) and high-value bits (PC4-7). Pin D4 is responsible for determining the state of the four high-value bits, and D0 determines the state of the four low-value bits.

**D2:** Determines the operating mode of PORTB, which is set to 0 here because state 1 is used for HANDSHAKING, which is not applicable in this project.

**D1:** Determines whether PORTB is input or output.

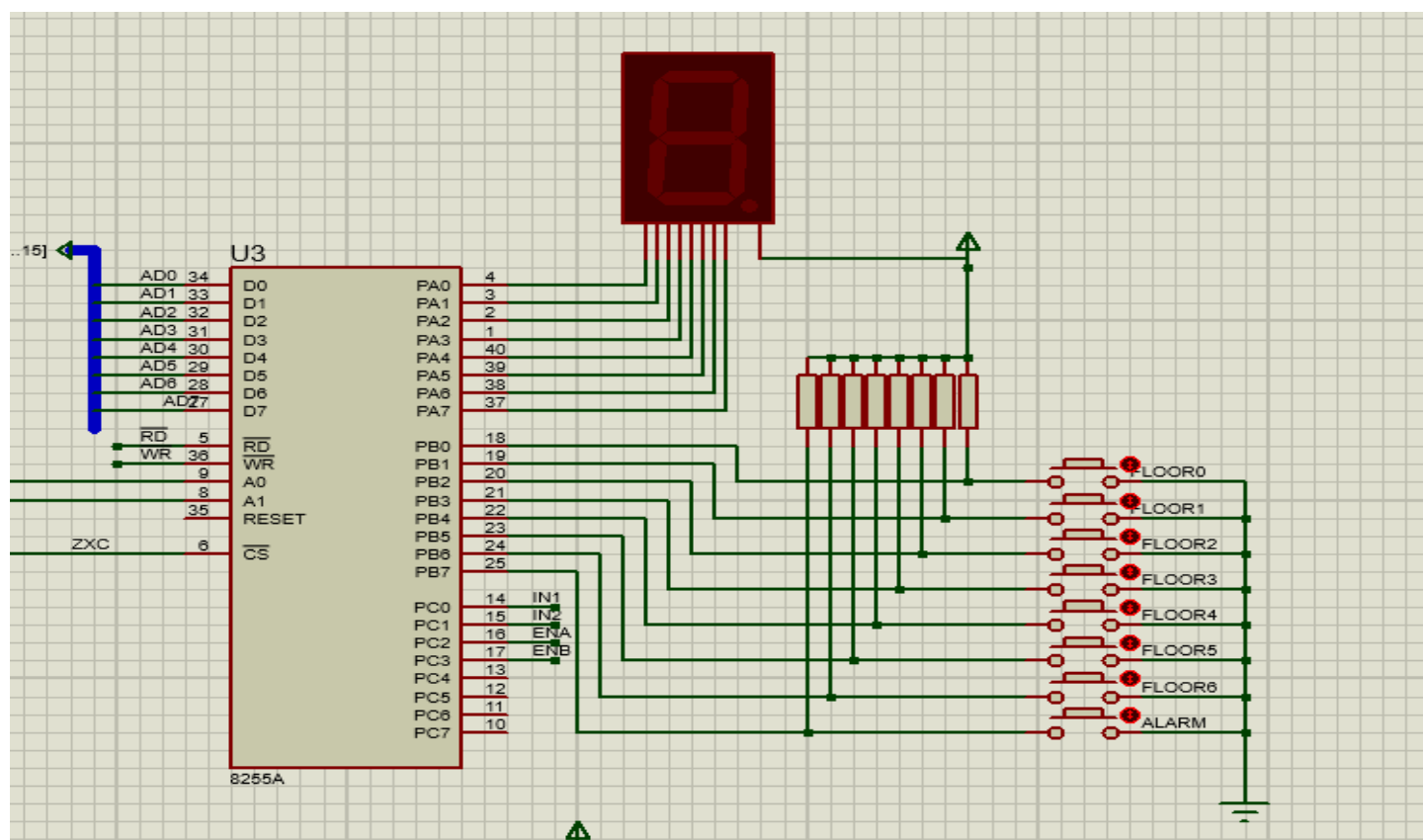
**A0\_A1:** These 2 pins are used to activate the ports in 8255, which have 4 states:

**00->** Activating PORTA

**01->** Activating PORTB

**10->** Activating PORTC

**11->** Activating the control pin



**This section has been designed to correctly pick up the valid address when data is placed on the data bus. For this purpose, a fully decoding system has been used. We must follow the steps we learned in class for decoding addresses.**

**1.In this system, valid addresses are equal to (00, 02, 04, 06). When these are given, we notice that their eight most significant bits are equal to zero, which we connect to our first latch. Then, by taking a NAND of its outputs, we create a logic one that forms one of the enable pins of our decoder.**

**2.In our eight least significant bits, pins D3-7 and D0 also generate zero, which due to the active low nature of the enable pin, an OR is taken from them.**

**3.The other enable pin is connected to the M/IO pin of the 8086 processor.**

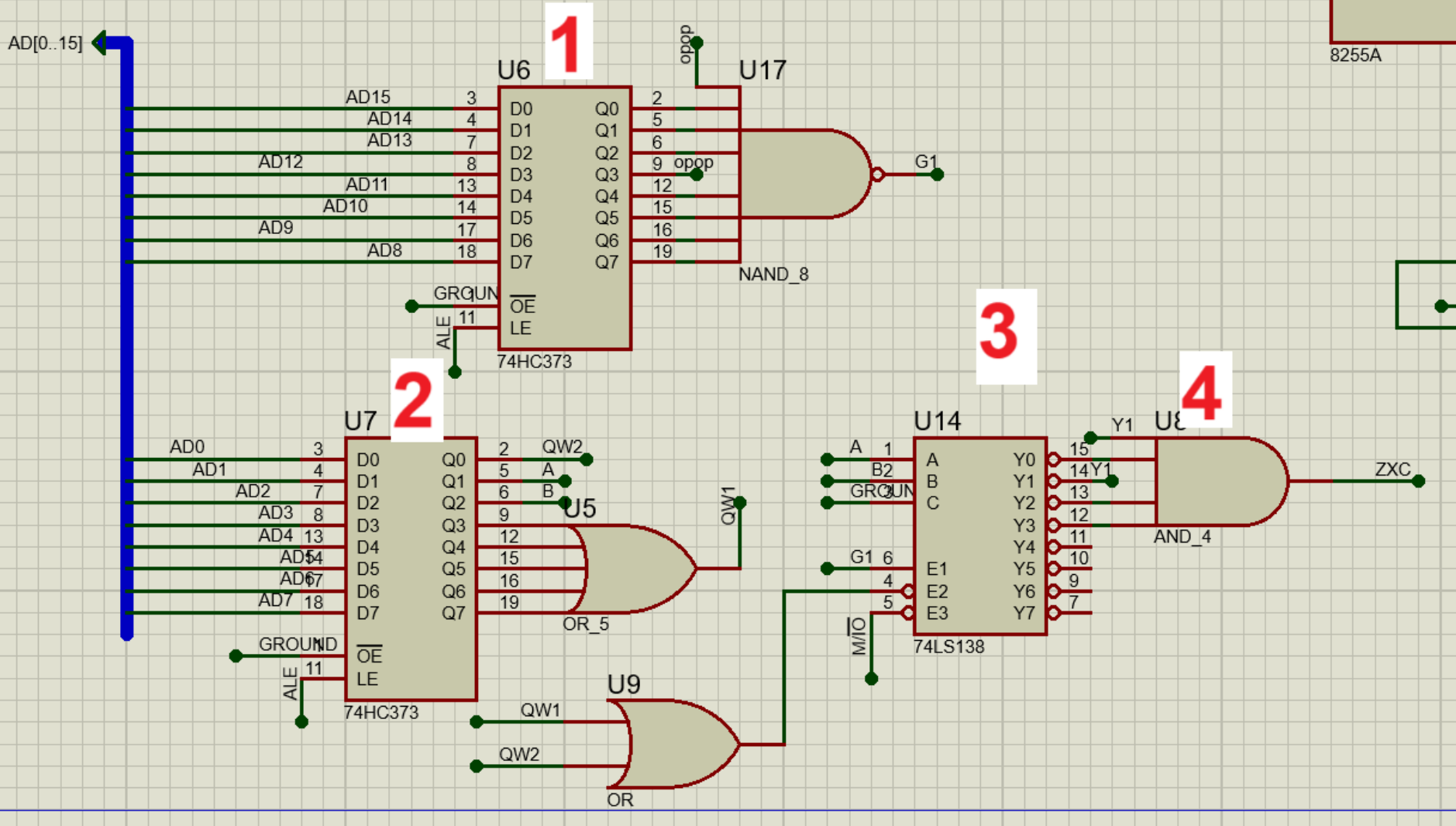
**4.Pins D1, D2 form our valid address decoder by connecting to lines A, B, and the remaining C line is responsible for activating and deactivating the four least significant and four most significant bits. Since we only have 4 addresses, we connect the C pin to the ground so that when one of the four valid addresses is placed, one of the 4 upper lines is activated.**

**After the CS pin is activated with the address 06H, our control pin becomes active and specifies which of the ports (A, B, C) should be used as output or input through the D lines, which have been fully explained.**

### **Control Line Settings Elevator Project**

**D7,D6,D5,D4,D3,D2,D1,D0**

**10000010**



## 8086 CODE

### DATA SEGMENT

PORTA EQU 00H

PORTB EQU 02H

PORTC EQU 04H

PORT\_CON EQU 06H

feli DW 0h

maghsad DW 0h

Table7Seg DB 0c0h, 0f9h, 0a4h, 0b0h, 099h, 092h, 082h, 088h

HERE WE DEFINE LOOKUP TABLE FOR DISPLAY NUMBERS IN 7SEGMENT;

DATA ENDS

### CODE SEGMENT

ASSUME DS:DATA ;

USE DS AS DEFAULT SEGMENT;

START:

MOV DX, PORT\_CON

MOV AL, 082h

OUT DX, AL

ACTIVATING CONTROL PORT;

MOV AL, 0c0h

MOV DX, PORTA

OUT DX, AL

SHOWING ZERO IN 7SEGMENT;

click:

```
MOV DX, PORTB
IN AL, DX
```

THIS FUNCTION COMPARE THE BUTTON INPUT  
IN ORDER TO UNDERSTAND WHICH BUTTON IS USED;

```
CMP AL, 11111110B
JZ F0
CMP AL, 11111101B
```

```
JZ F1
CMP AL, 11111011B
JZ F2
CMP AL, 11110111B
JZ F3
CMP AL, 11101111B
JZ F4
CMP AL, 11011111B
JZ F5
CMP AL, 10111111B
JZ F6
CMP AL, 01111111B
JZ ALARM
JMP click
```

F0:

```
MOV SI, OFFSET maghsad
MOV BX, 0
```

COMPARING THE MAGSHAD AND FELI STATE TO UNDERSTAND  
THAT ELEVATOR MUST GO DOWN OR UP;  
OTHER LEVEL ARE NOT WRITTEN TO COMPACT THE CODE SIZE

```
MOV [SI], BX
MOV SI, OFFSET feli
MOV CX, [SI]
CMP BX, CX
JG L6
CMP BX, CX
JL L8
```

```
MOV AL, 11000000B
MOV DX, PORTA
OUT DX, AL
JMP click
```



ALARM:

```
    mov dx,portc
    mov al,00001000B    TURN ON THE LED;
    out dx,al
    MOV AL, 088h    HERE WE PUSH CX AND MAKE A INTERNAL DELAY FUNCTION;
    PUSH CX
    MOV CX, 05FH
    MOV DX, PORTA
    out dx,ax
D2:
    nop
    LOOP D2
                                A LOOP TO CONSUME SPEND SOME TIME IN ALARM MODE;

    POP CX
```

```
    MOV SI, OFFSET feli
    MOV aX, [SI]
    MOV SI, OFFSET Table7Seg
    ADD SI, ax    SHOWING FELI STATE IN 7SEGMENT AND START THE
    mov ax,[si]    AGAIN;
    OUT DX, Ax
    mov dx,portc
    mov al,00000000B    WITH THIS INSTRUCTION WE DISABLE MOTOR ELEVATOR ;
                                AND ALARA LED THAT CONNECTED TO PORT_C;

    out dx,al
    JMP click

    JMP START
```

```
L6:    THIS FUNCTION IS FOR UNDRESTDANDING HOW MUCH WE MUST ;
    mov dx,portc    GO DOWN BY USING SUB METHOD;
    mov al,00000110B    HERE WE TURN ON THE ELEVATOR MOTOR AND ELEVATOR;
                                STARTS GOING DOWN;

    out dx,al
    MOV SI, OFFSET maghsad
    MOV BX, [SI]
    MOV SI, OFFSET feli
    MOV CX, [SI]
    SUB BX, CX
    XCHG BX, CX
```

L7: GOING DOWN STEP BY STEP USING A LOOP;

```
CALL DELAY    WE WILL DISCUSS ABOUT THIS FUNCTION LATER;
INC BX
MOV SI, OFFSET Table7Seg
ADD SI, bx
MOV AL, [SI]
MOV DX, PORTA
OUT DX, AL    WE DISPLAY CURRENT FLOOR IN 7SEGMENT;
MOV SI, OFFSET feli    HERE WE UPDATE VALUE OF FELI ;
MOV ax, [SI]
inc ax
mov [si],ax
LOOP L7
    mov dx,portc
mov al,00000000B    WE ARRIVED TO THE DESIRED FLOOR AND DISABLE MOTOR;
out dx,al
JMP click
```

L8: HOW MUCH WE MUST GO UP BY COMPARING MAGSHA AND FELI STATE  
USING ADD METHOD;

```
mov dx,portc
mov al,00000101B    HERE WE TURN ON THE ELEVATOR MOTOR AND ELEVATOR;
out dx,al    STARTS GOING UP;
MOV SI, OFFSET maghsad
MOV BX, [SI]    HERE WE LOAD MAGHSAD IN BX REGISTOR;
MOV SI, OFFSET feli
MOV CX, [SI]    HERE WE LOAD FELI IN CX REGISTOR;
SUB cx,bx    WE CALCULATE DIFFERENCES BETWEEN FLOORS AND SAVE VALUVE IN CX FOR;
    USING IT IN LOOP;
MOV SI, OFFSET feli    HERE WE LOAD FELI IN BX REGISTOR AGAIN FOR USING IN L9 LABLE;
MOV bx, [SI]
```

L9: GOING UP STEP BY STEP USING A LOOP;

```
call delay
MOV SI, OFFSET feli
MOV ax, [SI]    WE HAVE CURRENT FLOOR IN AX;
dec ax    WE REDUSE THE CURRENT FLOOR;
mov [si],ax    WE UPDATE VALUE OF FELI;
MOV SI, OFFSET Table7Seg
ADD SI, ax
```

```
MOV AL, [SI]
MOV DX, PORTA    WE DISPLAY THE CURRENT FLOOR IN 7SEGMENT;
OUT DX, AL
LOOP L9    THIS LOOP RUNS AS MUCH AS THE DIFFERENCES BETWEEN THE FLOORS;
    mov dx,portc
mov al,00000000B    WE DISABLE MOTOR;
out dx,al
JMP click
```

EXPLAINOF DELAY FUNCTIONS;

DELAY PROC NEAR

PUSH CX

MOV CX, 01FH

D1: HERE WE READ PORT\_B EVERY MOMENTS AND IF THE BUTTON WAS THE ALARM WE STOP;  
THE ELEVATOR ;

MOV DX, PORTB

in al,dx

cmp al,07fh

jz ALARM

LOOP D1

POP CX

RET WE MUST RETURN FROM THIS SUB FUNCTION;

DELAY ENDP

CODE ENDS END OF THE PROJECT;

END START