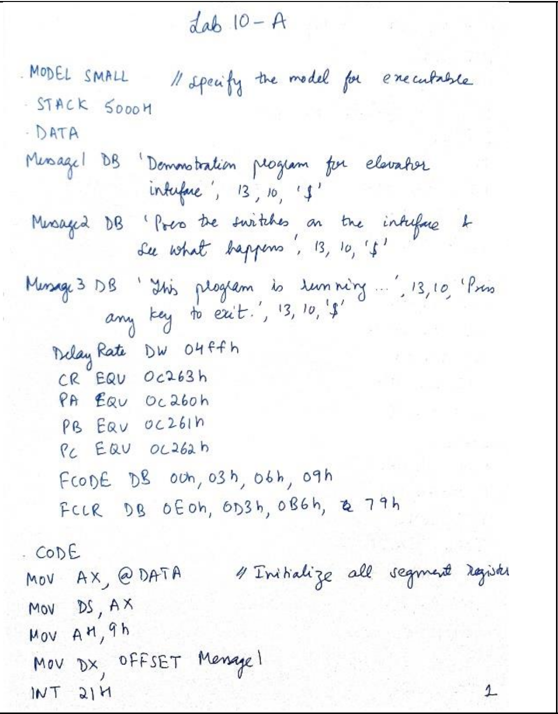
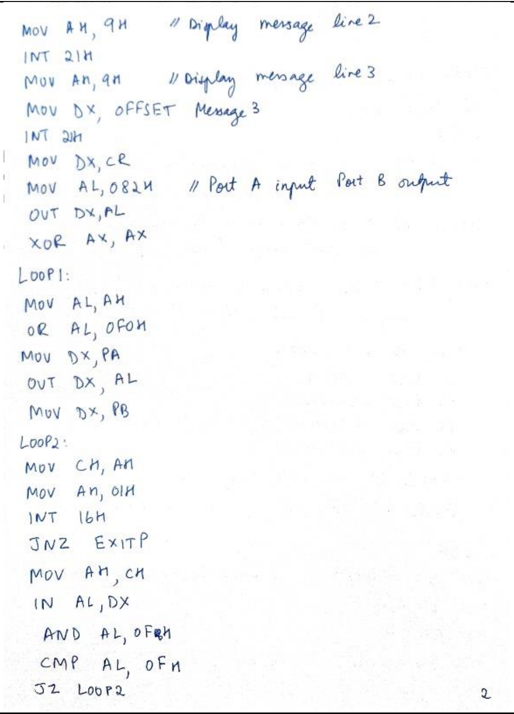
2019A7PS0020U Dhruv Maheshwari

Lab 10

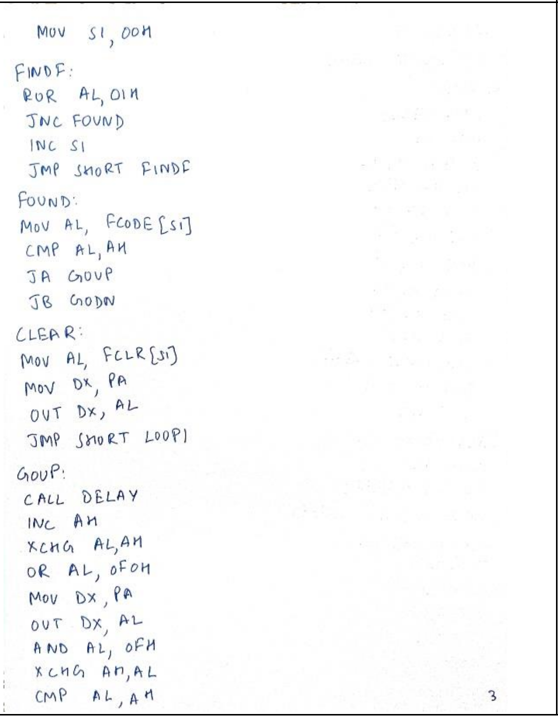
P1: To Interface an elevator to 8086 Microprocessor through 8255 Programmable Peripheral Interface (PPI).



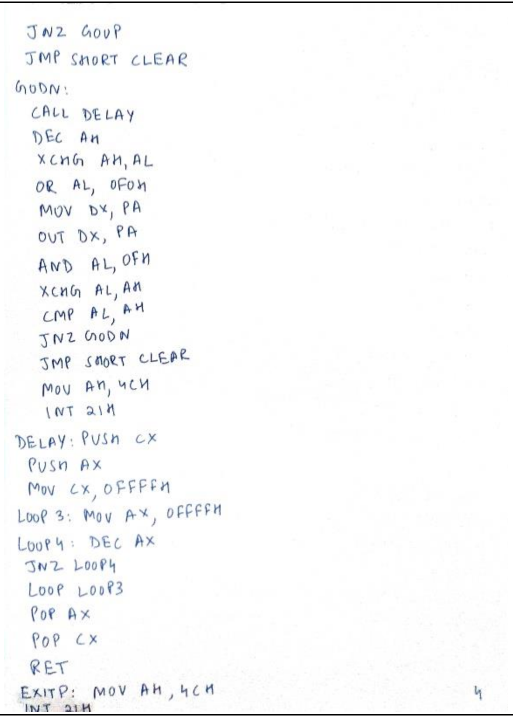
```
Lab 10 - A

.MODEL SMALL      // specify the model for executable
.STACK 5000H
.DATA
Message1 DB 'Demonstration program for elevator
            interface', 13, 10, '$'
Message2 DB 'Press the switches on the interface &
            see what happens', 13, 10, '$'
Message3 DB 'This program is running ...', 13,10, 'Press
         any key to exit.', 13, 10, '$'
  Delay Rate DW 04ffh
  CR EQU 0c263h
  PA EQU 0c260h
  PB EQU 0c261h
  Pc EQU 0c262h
  FCODE DB 00h, 03h, 06h, 09h
  FCLR DB 0E0h, 0D3h, 0B6h, & 79h

.CODE
MOV AX, @DATA        // Initialize all segment registers
MOV DS, AX
MOV AH, 9h
MOV DX, OFFSET Message1
INT 21H                                            1
```



```
MOV AH, 9H      // Display message line 2
INT 21H
MOV AH, 9H      // Display message line 3
MOV DX, OFFSET Message 3
INT 21H
MOV DX, CR
MOV AL, 082H    // Port A input Port B output
OUT DX, AL
XOR AX, AX
LOOP1:
MOV AL, AH
OR AL, 0F0H
MOV DX, PA
OUT DX, AL
MOV DX, PB
LOOP2:
MOV CH, AH
MOV AH, 01H
INT 16H
JNZ EXITP
MOV AH, CH
IN AL, DX
 AND AL, 0FH
 CMP AL, 0FH
 JZ LOOP2                                          2
```



```
 MOV SI, 00H
FINDF:
ROR AL, 01H
JNC FOUND
INC SI
JMP SHORT FINDF
FOUND:
MOV AL, FCODE[SI]
CMP AL, AH
JA GOUP
JB GODN
CLEAR:
MOV AL, FCLR[SI]
MOV DX, PA
OUT DX, AL
JMP SHORT LOOP1
GOUP:
CALL DELAY
INC AH
XCHG AL,AH
OR AL, 0F0H
MOV DX, PA
OUT DX, AL
AND AL, 0FH
XCHG AH,AL
CMP AL, AH                                         3
```

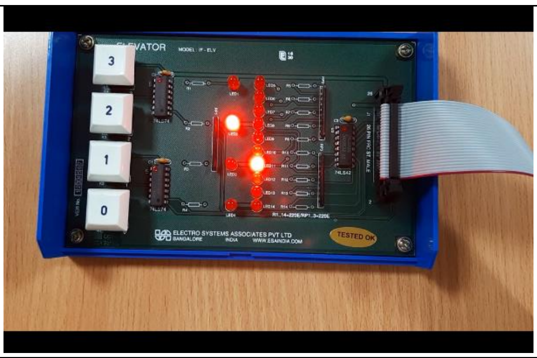
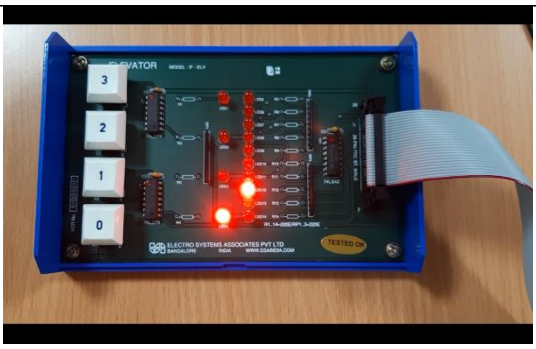


```
JNZ GOUP
JMP SHORT CLEAR
GODN:
CALL DELAY
DEC AH
XCHG AH,AL
OR AL, 0F0H
MOV DX, PA
OUT DX, PA
AND AL, 0FH
XCHG AL, AH
CMP AL, AH
JNZ GODN
JMP SHORT CLEAR
MOV AH, 4CH
INT 21H
DELAY: PUSH CX
PUSH AX
MOV CX, 0FFFFH
LOOP 3: MOV AX, 0FFFFH
LOOP 4: DEC AX
JNZ LOOP4
LOOP LOOP3
POP AX
POP CX
RET
EXITP: MOV AH, 4CH                                  4
INT 21H
```

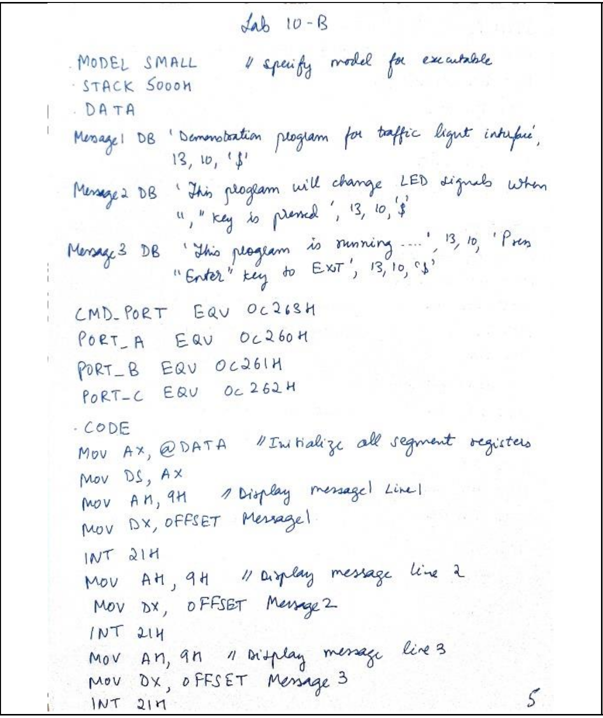
Observation:

The 10 lights represent the motion of the elevator. The delay between switching off one LED and turning on adjacent LED represents the speed of the lift. The service request is read through flip flops of port B. The corresponding light for the key pressed lights up and is reset (switches off) when elevator comes up to that level (in other words when flip flops are reset).

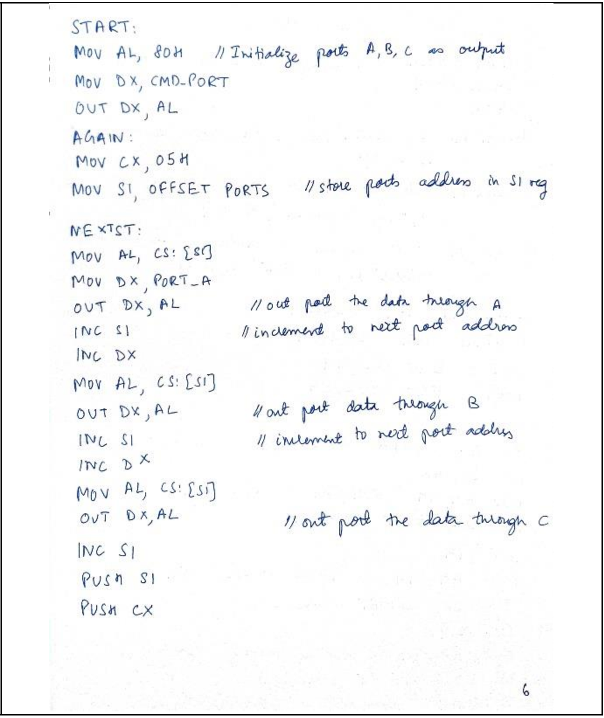
Output:

	
Lift going from 0 to 3	Lift going from 3 to 0

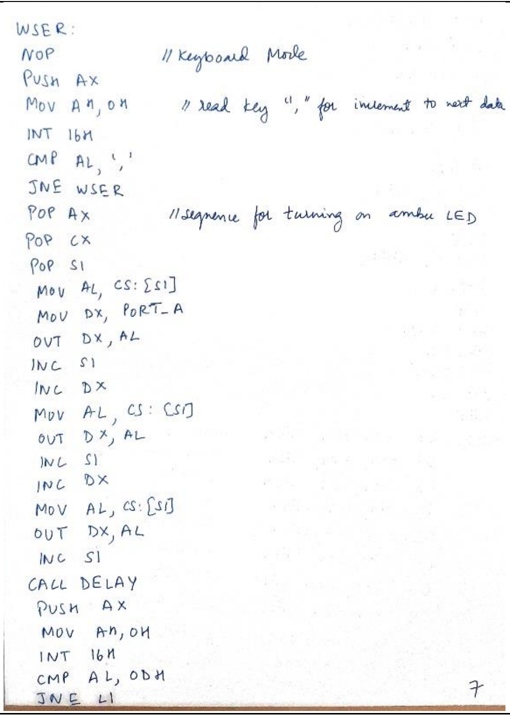
P1: To Interface a traffic light to 8086 Microprocessor through 8255 Programmable Peripheral Interface (PPI).



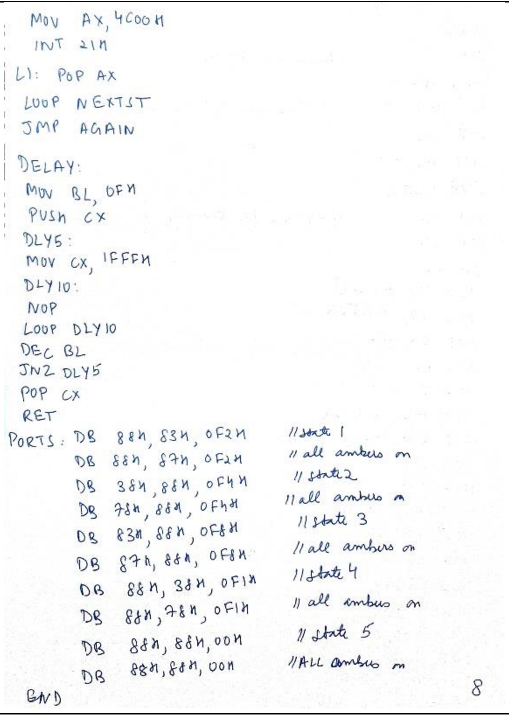
```
                    Lab 10-B

.MODEL SMALL        // specify model for executable
.STACK 5000H
.DATA
Message1 DB 'Demonstration program for traffic light interface',
            13, 10, '$'
Message2 DB 'This program will change LED signals when
            "," key is pressed', 13, 10,'$'
Message3 DB 'This program is running....', 13, 10, 'Press
            "Enter" key to EXIT', 13, 10, '$'

CMD_PORT  EQU 0C263H
PORT_A    EQU 0C260H
PORT_B    EQU 0C261H
PORT_C    EQU 0C262H

.CODE
MOV AX, @DATA    // Initialize all segment registers
MOV DS, AX
MOV AH, 9H       // Display message1 Line1
MOV DX, OFFSET Message1
INT 21H
MOV AH, 9H       // Display message line 2
MOV DX, OFFSET Message2
INT 21H
MOV AH, 9H       // Display message line 3
MOV DX, OFFSET Message 3
INT 21H
                                                  5
```



```
START:
MOV AL, 80H     // Initialize ports A, B, C as output
MOV DX, CMD_PORT
OUT DX, AL
AGAIN:
MOV CX, 05H
MOV SI, OFFSET PORTS     // store ports address in SI reg

NEXTST:
MOV AL, CS: [SI]
MOV DX, PORT_A
OUT DX, AL            // out port the data through A
INC SI                // increment to next port address
INC DX
MOV AL, CS: [SI]
OUT DX, AL            // out port data through B
INC SI                // increment to next port address
INC DX
MOV AL, CS: [SI]
OUT DX, AL               // out port the data through C
INC SI
PUSH SI
PUSH CX
                                                  6
```



```
WSER:
NOP               // Keyboard Mode
PUSH AX
MOV AH, 0H        // read key "," for increment to next data
INT 16H
CMP AL, ','
JNE WSER
POP AX            // sequence for turning on amber LED
POP CX
POP SI
 MOV AL, CS:[SI]
 MOV DX, PORT_A
 OUT DX, AL
 INC SI
 INC DX
 MOV AL, CS:[SI]
 OUT DX, AL
 INC SI
 INC DX
 MOV AL, CS:[SI]
 OUT DX, AL
 INC SI
CALL DELAY
 PUSH AX
 MOV AH, 0H
 INT 16H
 CMP AL, 0DH
 JNE L1
                                         7
```



```
 MOV AX, 4C00H
 INT 21H
L1: POP AX
 LOOP NEXTST
 JMP AGAIN

DELAY:
 MOV BL, 0FH
 PUSH CX
 DLY5:
 MOV CX, 1FFFH
 DLY10:
 NOP
 LOOP DLY10
 DEC BL
 JNZ DLY5
 POP CX
 RET
PORTS: DB 88H, 83H, 0F2H      // state 1
       DB 88H, 87H, 0F2H      // all ambers on
       DB 38H, 88H, 0F4H      // state2
       DB 78H, 88H, 0F4H      // all ambers on
       DB 83H, 88H, 0F8H      // state 3
       DB 87H, 88H, 0F8H      // all ambers on
       DB 88H, 38H, 0F1H      // state 4
       DB 88H, 78H, 0F1H      // all ambers on
       DB 88H, 88H, 00H       // state 5
       DB 88H, 88H, 00H       // ALL ambers on
END                                      8
```

Observation:

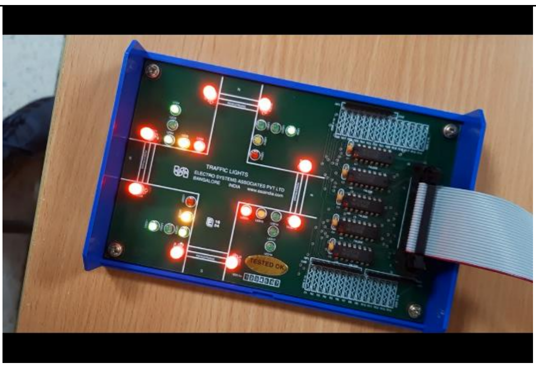
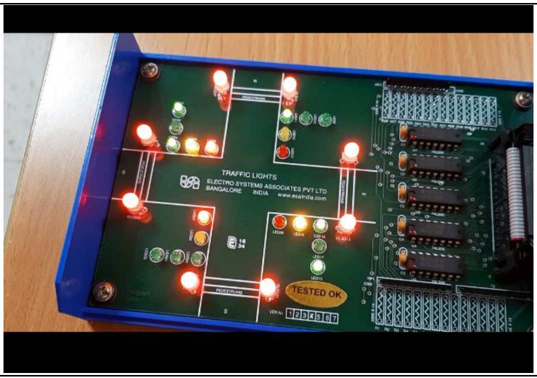
This board has 6 LED lights at each of its 4 corners; total 24, all ports are used as output ports. The six lights are Red, Amber, left, right, straight and pedestrian.

First five LEDs glow when they receive active high input and turn off on active low. Pedestrian light glows red on active high and green on active low.

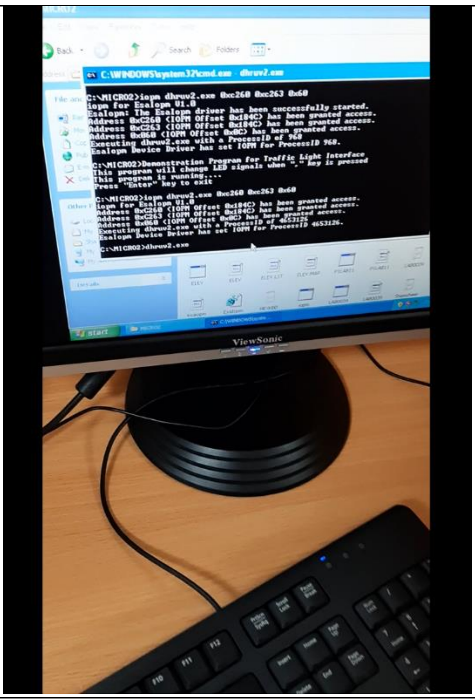
The pedestrian light for a particular corner would only glow red when left right and centre are green for that corner and would glow green otherwise. A pedestrian should only cross when two adjacent pedestrian lights are glowing green and wait otherwise.

The traffic coming from one direction is guided by the lights in the opposite direction.

Output:

2019A7PS0020U Dhruv Maheshwari



C:\WINDOWS\system32\cmd.exe - dhruv2.exe

```
C:\MICRO2>iopm dhruv2.exe 0xc260 0xc263 0x60
iopm for Esalopm V1.0
Esalopm: The Esalopm driver has been successfully started.
Address 0xC260 (IOPM Offset 0x184C) has been granted access.
Address 0xC263 (IOPM Offset 0x184C) has been granted access.
Address 0x060 (IOPM Offset 0x0C) has been granted access.
Executing dhruv2.exe with a ProcessID of 968
Esalopm Device Driver has set IOPM for ProcessID 968.

C:\MICRO2>Demonstration Program for Traffic Light Interface
This program will change LED signals when "." key is pressed
This program is running....
Press "Enter" key to exit

C:\MICRO2>iopm dhruv2.exe 0xc260 0xc263 0x60
iopm for Esalopm V1.0                           has been granted access.
Address 0xC260 (IOPM Offset 0x184C) has been granted access.
Address 0xC263 (IOPM Offset 0x184C) has been granted access.
Address 0x060 (IOPM Offset 0x0C) has been granted access.
Executing dhruv2.exe with a ProcessID of 4653126
Esalopm Device Driver has set IOPM for ProcessID 4653126.

C:\MICRO2>dhruv2.exe
```