1

Consider the following table contains the address and object file for an 8051 assembly program and,

(i) find the corresponding HEX file to load into the ROM location of 8051 where *W, X, Y, and Z* represent the last four digits of your ID.

(ii) also verify the HEX file, if the data in line number 9 have been changed to F5XW.

Table 1: Object file and address of an assembly program

Line Number	Address (HEX)	Object File (HEX)
01	0000	7455
02	0002	F590
03	0004	7C99
04	0006	7D67
05	0008	120012
06	000B	74AA
07	000D	F590
08	000F	120012
09	0012	F5WX
10	0014	80EC
11	0016	C004
12	0018	C005
13	001A	7CFF
14	001C	7DFF
15	001E	DDFE
16	001F	DCFA
17	0021	D005
18	0023	D004
19	0025	00YZ

2

Solve the following problem using Assembly Language

Simple Calculate

In this problem, the task is to read a code of a product 1, the number of units of product 1, the price for one unit of product 1, the code of a product 2, the number of units of product 2 and the price for one unit of product 2. After this, calculate and show the amount to be paid.

Input

The input file contains two lines of data. In each line there will be 3 values: three integers.

Output

The output file must be a message like the following example where "Valor a pagar" means **Value to Pay**. Remember the space after ":" and after "R$" symbol. The value must be presented with 2 digits.

Input Samples	Output Samples
7 5 1 6 2 2	VALOR A PAGAR: R$ 09
9 2 2 4 2 2	VALOR A PAGAR: R$ 08
5 2 3 8 3 1	VALOR A PAGAR: R$ 09

Solution:

```
.data

message1 db 'VALOR A PAGAR: R$'

.code
BUBT PROC
    ;initialize data segment
    MOV AX, @DATA
    MOV DS,AX
    ;input code number of the product
    MOV AH,1
    int 21h

    MOV AH,2
    MOV DL, 20h
```

```
 int 33
 MOV DL, 20h
 int 33

 ;input number of units of product
 MOV AH,1
 int 21h
 MOV BL,AL
 SUB BL,30h; convert hexa to decimal

 MOV AH,2
 MOV DL, 20h
 int 33
 MOV DL, 20h
 int 33
 ;input number as price per unit of product
 MOV AH,1
 int 21h
 MOV BH,AL
 SUB BH,30h; convert hexa to decimal


;print a new line

MOV AH,2
MOV DL,10
int 21h
MOV DL,13
int 21h



 ;repeat same process for second product

  ;input code number of the product
 MOV AH,1
 int 21h

 MOV AH,2
 MOV DL, 20h
 int 33
 MOV DL, 20h
 int 33

 ;input number of units of product
 MOV AH,1
 int 21h
 MOV CL,AL
 SUB CL,30h; convert hexa to decimal

 MOV AH,2
 MOV DL, 20h
 int 33
 MOV DL, 20h
 int 33
 ;input number as price per unit of product
 MOV AH,1
 int 21h
 MOV CH,AL
 SUB CH,30h; convert hexa to decimal



 ;multiply

 MOV AX,0
 MOV AL,BL
 MUL BH

 MOV BL,AL
 ;ADD BL,30h;convert decimal to hexa


 MOV AX,0
 MOV AL,CL
 MUL CH

 MOV CL,AL
 ;ADD CL, 30h;;convert decimal to hexa


 ADD BL,CL
 ADD BL,30h  ;convert decimal to hexa


;print a new line

MOV AH,2
MOV DL,10
int 21h
MOV DL,13
int 21h

MOV AH,9
LEA DX,message1
int 21h
```

```
  MOV AH,2
  MOV DL,24h
  int 21h

  MOV AH,2
  MOV DL,20h
  int 21h



  MOV AH,2
  MOV DL,30h
  int 21h

  MOV DL,BL
  int 21h



   BUBT ENDP
END BUBT
```

3

Solve the following problem using Assembly Language

Difference

Read four integer values named A, B, C and D. Calculate and print the difference of product A and B by the product of C and D (A * B - C * D).

Input

The input file contains 4 integer values.

Output

Print **DIFERENCA** (DIFFERENCE in Portuguese) with all the capital letters, according to the following example, with a blank space before and after the equal signal.

Input Samples	Output Samples
5 1 4 1	DIFERENCA = 1
3 3 2 2	DIFERENCA = 5
6 1 5 1	DIFERENCA = 1

Solution:

```
.data
message1 db 'DIFERENCA = $'


.code
BUBT PROC
    ;initialize data segment
    MOV AX, @DATA
    MOV DS,AX

    MOV AH,1
```

```
int 21h
MOV BL,AL
SUB BL,30h

MOV AH,2
MOV DL, 10
int 33
MOV DL, 13
int 33



MOV AH,1
int 21h
MOV BH,AL
SUB BH,30h

MOV AH,2
MOV DL, 10
int 33
MOV DL, 13
int 33




MOV AH,1
int 21h
MOV CL,AL
SUB CL,30h

MOV AH,2
MOV DL, 10
int 33
MOV DL, 13
int 33
```

```
  MOV AH,1
  int 21h
  MOV CH,AL
  SUB CH,30h

  MOV AH,2
  MOV DL, 10
  int 33
  MOV DL, 13
  int 33



 ; multiply
 MOV AX,0
 MOV AL,BL
 MUL BH
 MOV BL,AL



MOV AX,0
 MOV AL,CL
 MUL CH
 MOV CL,AL

;subtraction

SUB BL,CL
ADD BL,30H
```

```
   ;print a new line

   MOV AH,2
   MOV DL,10
   int 21h
   MOV DL,13
   int 21h



   ;print message
   MOV AH,9
   LEA DX,message1
   int 21h



   ;print space
   MOV AH,2
   MOV DL,20h
   int 21h


   ;print result
   MOV DL,BL
   int 21h


    BUBT ENDP
END BUBT
```

4 **Solve the following problem using Assembly Language**

Simple sum

Read two integer values, in this case, the variables A and B. After this, calculate the sum between them and assign it to the variable **SOMA**. Write the value of this variable.

Input

The input file contains 2 integer numbers.

Output

Print the variable **SOMA** with all the capital letters, with a blank space before and after the equal signal followed by the corresponding value to the sum of A and B. Like all the problems, don't forget to print the end of line, otherwise you will receive "Presentation Error"

Input Samples	Output Samples
3 1	SOMA = 4
9 0	SOMA = 9
0 5	SOMA = 5

Solution:

```
.data
message1 db 'SOMA = $'
.code
BUBT PROC
   ;initialize data segment
   MOV AX, @DATA
   MOV DS,AX
   MOV AH,1
   int 21h
   MOV BL,AL
```

```
  MOV AH,2
  MOV DL, 10
  int 33
  MOV DL, 13
  int 33
  MOV AH,1
  int 21h
  MOV BH,AL


  MOV AH,2
  MOV DL, 10
  int 33
  MOV DL, 13
  int 33



;Addition

ADD BL,BH
SUB BL,30H


 ;print a new line

 MOV AH,2
 MOV DL,10
 int 21h
 MOV DL,13
 int 21h

 ;print message
 MOV AH,9
 LEA DX,message1
 int 21h
 ;print space
 MOV AH,2
 MOV DL,20h
 int 21h


 ;print result
 MOV DL,BL
 int 21h


  BUBT ENDP
END BUBT
```

5 Assume that you have to build a sensor based automated locker system. When sensor detect any object movement the door will open otherwise it will be locked and also show message in the display device. You have to build this system using Arduino UNO & if necessary you can use PIR sensor, servo motor and Lcd display. Now **implement** the system through programming code with appropriate design.

Solution:

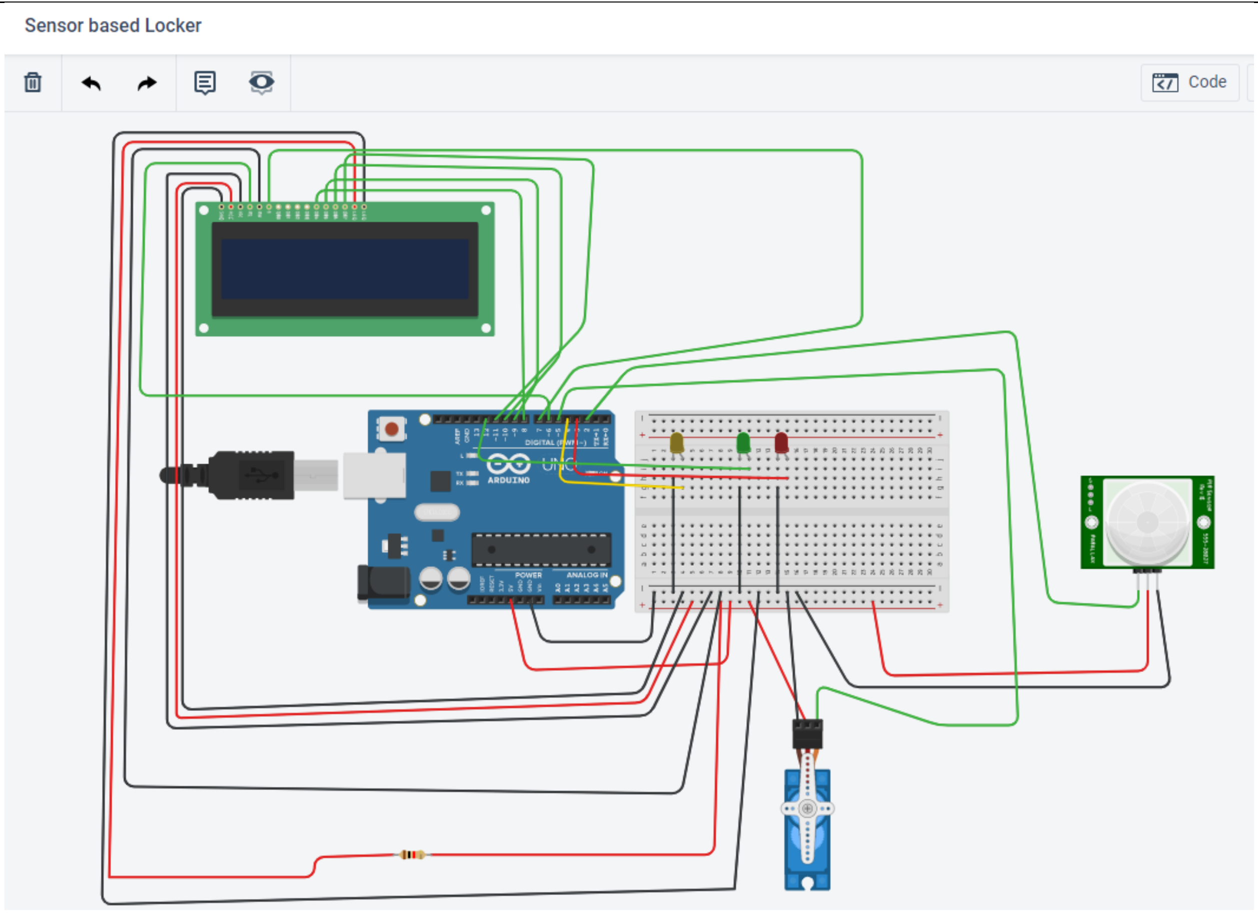
Code:

```
#include <LiquidCrystal.h>

#include <Servo.h>


#define ledG 12

#define ledR 3

#define ledY 4


LiquidCrystal lcd(6,7,8,9,10,11);
```

```
Servo object_servo; // servo object to control servo motor


int interrCount=0;
void setup()
{
  pinMode(ledG, OUTPUT);  // setup pin as output
  pinMode(ledR, OUTPUT);
  pinMode(ledY, OUTPUT);

 digitalWrite(ledR, LOW);  // clear led
  digitalWrite(ledG, LOW);
  digitalWrite(ledY, LOW);



Serial.begin(9600);
lcd.begin(16,2);
object_servo.attach(5);  // attach servo motor with arduino pin 9

  object_servo.write(0);
 attachInterrupt(0, interruptChange, RISING);
 // method to detyect obje

}
void loop()
{
   interrCount++;
  digitalWrite(ledR, HIGH);  // high red led


  digitalWrite(ledG, LOW);// low green led
  delay(300);
  digitalWrite(ledR, LOW); // low red led
  digitalWrite(ledG, HIGH);  // high green led
  delay(300);


if(interrCount == 10)
{
    interrCount =0;
    digitalWrite(ledY, LOW);  // low yellow led


  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("Object-Absent");
  lcd.setCursor(0,1);
  lcd.print("Door Close");


   delay(500);
  object_servo.write(0);
  //sets the servo position according to the scaled value


}
```

```
}
void  interruptChange()
{


  digitalWrite(ledY, HIGH);
  // high yellow led that indicate movement
          // of object
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("Object Detect ");
  lcd.setCursor(0,1);
  lcd.print("Door Open");
  object_servo.write(90);
   delay(500);
  //sets the servo position according to the scaled value




}
```

Design:

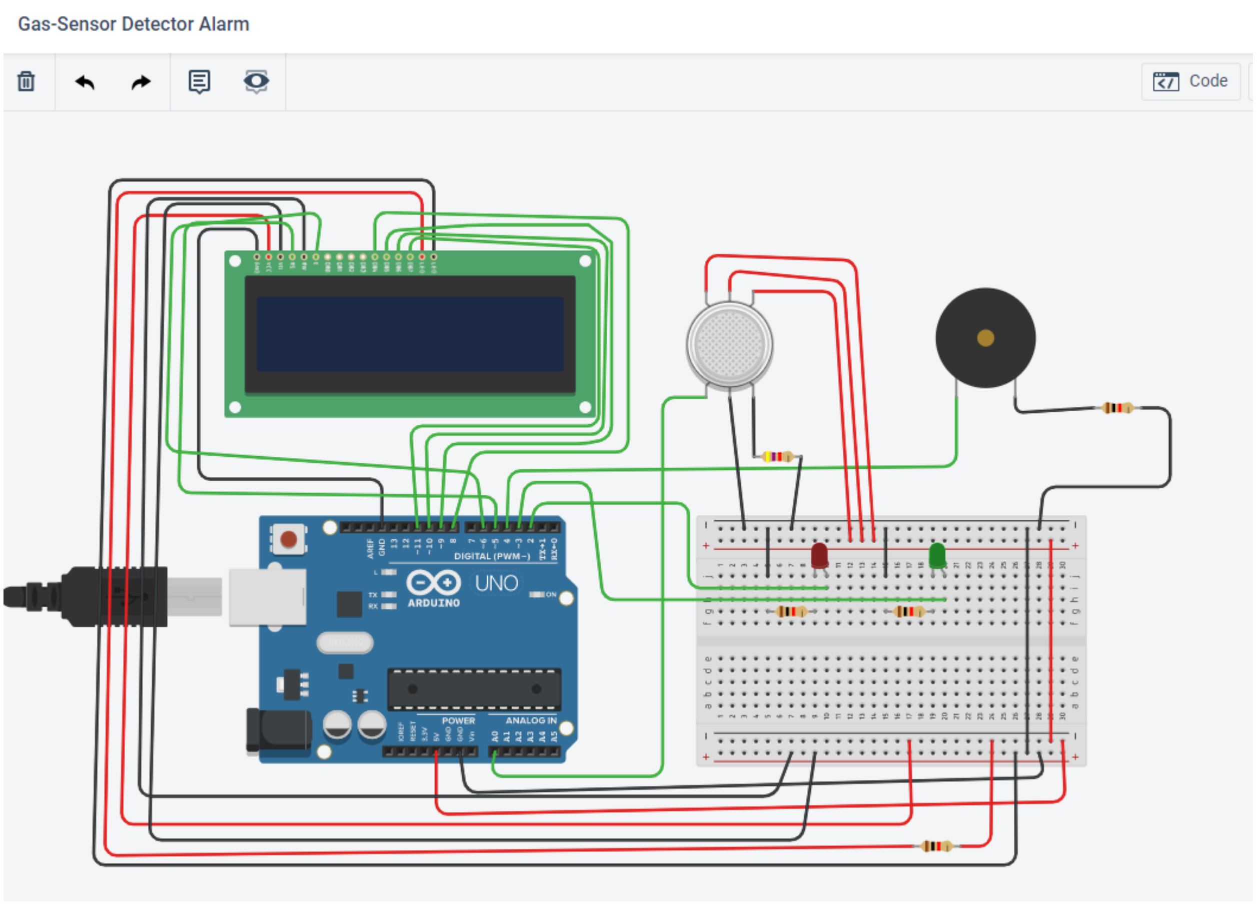


6 Assume that you have to build a Gas sensor based automated alarming system to skip unwanted danger situation. When sensor detect any dangerous amount of gas it will start alarm through on a buzzer otherwise alarm will be off as well as also show message in the display device. You have to build this system using Arduino UNO & if necessary you can use smoke detection sensor, buzzer and LCD display. Now **implement** the system through programming code with appropriate design.

Solution:

Code:

```
#include <LiquidCrystal.h>

LiquidCrystal lcd(5, 6, 8, 9, 10, 11);



int redLed= 2;

int greenLed= 3;

int buzzer =4;
int sensor=A0;
int sensorThreshHold=400;



void setup()
{
  pinMode(redLed, OUTPUT);
  pinMode(greenLed, OUTPUT);
  pinMode(buzzer, OUTPUT);
  pinMode(sensor, INPUT);
  Serial.begin(9600);
  lcd.begin(16,2);




}

void loop()
{
  int analogvalue = analogRead(sensor);
  Serial.println(analogvalue);

  if(analogvalue > sensorThreshHold)
  {
    digitalWrite(redLed, HIGH);
     digitalWrite(greenLed, LOW);
    tone(buzzer, 1000, 10000);
    lcd.clear();
    lcd.setCursor(0,1);
    lcd.print("ALERT");
    delay(1000);
    lcd.clear();
    lcd.setCursor(0,1);
    lcd.print("EVACUATE");
     delay(1000);



  }
  else
  {
    digitalWrite(redLed, LOW);
     digitalWrite(greenLed, HIGH);
    noTone(buzzer);
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("SAFE");
    delay(1000);
    lcd.clear();
    lcd.setCursor(0,1);
    lcd.print("ALL CLEAR");
     delay(1000);
  }




}
```

	Design:  Gas-Sensor Detector Alarm Code DIGITAL (PWM~) UNO ARDUINO POWER ANALOG IN
7	Assume that you have to build a password based automated locker system. When user input correct password the door will open otherwise it will be locked and also show message in the display device. You have to build this system using Arduino UNO & if necessary you can use Keypad, servo motor and Lcd display. Now **implement** the system through programming code with appropriate design. Solution: Password based door locker system Code: // C/ C++ code

```
#include<EEPROM.h>

#include<Servo.h>

#include<LiquidCrystal.h>

#include<Keypad.h>


const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;

LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

Servo myservo;

char keypressed;

char code[]= {'1','2','3','4'};


short a=0,i=0,s=0,j=0; //Variables used later


const byte ROWS = 4; //four rows

const byte COLS = 4; //four columns

//define the cymbols on the buttons of the keypads

char hexaKeys[ROWS][COLS] = {

  {'1','2','3','A'},

  {'4','5','6','B'},

  {'7','8','9','C'},

  {'*','0','#','D'}

};

byte rowPins[ROWS] = {10, 9, 8, 7}; //connect to the row pinouts of the keypad

byte colPins[COLS] = {A0, A1, A2, A3}; //connect to the column pinouts of the keypad


//initialize an instance of class NewKeypad
```

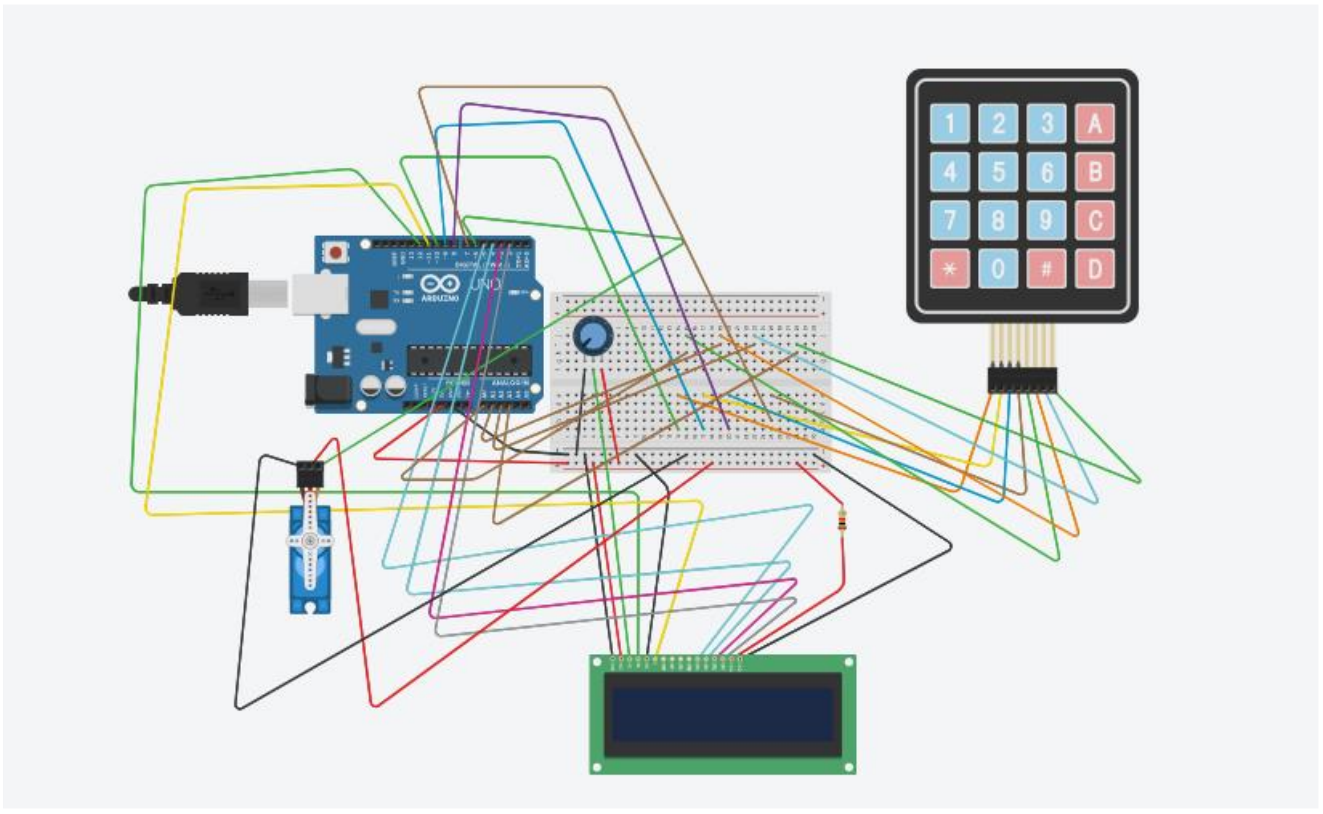
```
Keypad myKeypad = Keypad( makeKeymap(hexaKeys), rowPins, colPins, 4, COLS);


void setup(){
  //Serial.begin(9600);
  lcd.begin(16,2);
  myservo.attach(6);
  myservo.write(0);


            // User Manual
            lcd.clear();
            lcd.setCursor(0,0);
            lcd.print("Press * For ");
            lcd.setCursor(0,1);
            lcd.print("Input Password");
            delay(2000);


            lcd.clear();
            lcd.setCursor(0,0);
            lcd.print("After InputPress");
            lcd.setCursor(0,1);
            lcd.print("A Key forEnter");
            delay(2000);


    for(i=0 ; i<sizeof(code);i++){ //When you upload the code the first time keep it //commented
     EEPROM.get(i, code[i]); //Upload the code and change it to store it in the //EEPROM
 } //Then uncomment this for loop and reupload the code (It's done only once)
```

```
}// end setup method


void loop(){


// Door locker password based


 keypressed = myKeypad.getKey();
    if(keypressed == '*'){


      lcd.clear();
             lcd.setCursor(0,0);
             lcd.print("Enter password");
             ReadCode();
                    if(a==sizeof(code))  {
                      OpenDoor();
                     }
                    else{
                    lcd.clear();
                    lcd.print("Wrong password");
                    }
             delay(2000);
             lcd.clear();
             lcd.print("Standby");
             delay(3000);
             lcd.setCursor(0,0);
             lcd.print("Door-close");
```

```
  myservo.write(0);


        }

}


void ReadCode(){

       i=0;

       a=0;

       j=0;


     while(keypressed != 'A'){

           keypressed = myKeypad.getKey();

             if(keypressed != NO_KEY && keypressed != 'A' ){

              lcd.setCursor(j,1);

              lcd.print("*");

              j++;

            if(keypressed == code[i]&& i<sizeof(code)){

                 a++;

                 i++;

                 }

            else{

                a--;

            }

            }

     }

    keypressed = NO_KEY;
```

```
}


void OpenDoor(){
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("Welcome");
  myservo.write(90);
  lcd.setCursor(0,1);
  lcd.print("Door Unlock");
  delay(2000);


 }
```

Design:

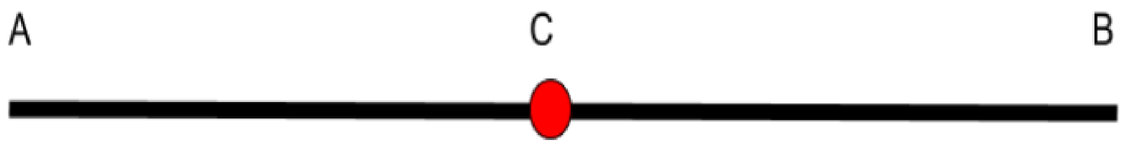


8 Consider the following table contains the address and object file for an 8051 assembly program and, find the corresponding HEX file to load into the ROM location of 8051.

Table 1: Object file and address of an assembly program

Line Number	Address (HEX)	Object File (HEX)
01	0000	7455
02	0002	F590
03	0004	7C99
04	0006	7D67
05	0008	120012
06	000B	74AA
07	000D	F590
08	000F	120012
09	0012	F523
10	0014	80EC
11	0016	C004
12	0018	C005
13	001A	7CFF
14	001C	7DFF
15	001E	DDFE
16	001F	DCFA
17	0021	D005
18	0023	D004
19	0025	00FE

9 **Assume that BUBT authority wants to introduce automated library management system. For this regard, as a student of CSE you have to build a book issue system, where a student can easily know if he/she needs a book that corresponding book is available or not. Firstly, you can think it as a simple scenario according to your CSE department, where available books are C, C++, Java, Data Structure, Algorithm, Database, Operating systems, Artificial Intelligence, Computer network, Compiler. Every book items must have a unique reference number. According to the aforementioned books, reference number may be '0' for C, '1' for C++, '2' for Java, & '3' for Data Structure and so on. When a student give a reference book number as input, he/she gets a number as the response of given input which indicates the corresponding book is available or not. The range of the number may be 0 to 45 where 0 means not available and rest of the numbers indicate the availability of that book. You can use port P2 to get input of the reference book item and send the availability information of the corresponding book item through port P1 as output, here also mention that the process will be working**

	continuously. Now implement **the system using 8051 Assembly language program. (Example: if the input to Port P2** **is 2, then output to port P1 is 25 which means that the available number of Java book is 25)**
10	Students of CSE 44 intake, BUBT wants to reduce road accident in Bangladesh by warning the driver that there is a red-spot (where frequently road accident occurred) when vehicles are around the red-spot. A sensor is connected with pin A2 Arduino/port2 of 8051 microcontroller which provides the location of the vehicle in an integer number. Let, consider a road from A to B where C is the black-spot and vehicles can move from A to B or B to A.  *Fig: Road from A to B where C is red-spot* When it moves around the red-spot C and less than or equal to five steps behind or beyond to C, then display “Around Red-spot” in an LCD display. When it is in the red-spot C, then display “In Red-spot” in the LCD display. Now design and implement the task using 8051 microcontroller /Arduino with code.
11	Consider the following Intel Hex file :1000000075805575905575A0557DFA111C**XY**80AA9F :100010007**XY**0AA75A0AA7DFA111C80E47C237C4F02 :07002000DBFEDCFADDF6**XY**35 :00000001FF Now, **analyze** the above HEX file and calculate the checksum byte for the first three lines as well as examine whether the information is corrupted or not. [Hints: Here, XY means last two digits of your ID (for example, if ID is 19202103433 then ‘XY’ will be ‘33’)] Solution: (The 8051 Microcontroller Book-chapter Hardware Connection and Intel HEX File) **Analysis of the HEX file for the first three line given below:** **:** here, colon symbol of each line indicate the starting of the line. **10H** value of the first two line indicates that these two line contains 16 bytes of data and **07H** value of the third line indicate that it contains 7 bytes of data.

0000H value of the first line, **0010H** value of the second line, and **0020H** value of the third line represent the starting address of the ROM from where the data or information bytes will be stored.

After this information there is also consider two digits like **00** for the first three line which indicate whether the loader will stop to store value in ROM or not. Here, **00** means loader should have to store more information after this line and **01** value of the fourth line indicates this is the last line and loader should have to stop loading information or code into ROM.

After that the remaining hexadecimal values except last byte of each three line indicates the actual data or information that is saved in ROM.

The last byte of each three lines like **9FH**, **02H**, and **35H** represent the checksum byte for the corresponding line.

CHECKSUM calculation for Error detection

Consider the first line,

Firstly, we add the hexadecimal information byte wise except the last byte. Here, XY will replace to the last two digit of the ID, like 33

10 +00+00+00+75+80+55+75+90+55+75+A0+55+7D+FA+11+1C+**33+**80+AA = **71FH**

If we discard the carry bit we get, 1FH

Now 2's complement of the 1FH (0001 1111) will be (1110 0001) or E1H.

But, the last byte is 9FH that is not matched.

So, the information of the first line is corrupted.

Continuing this process for the 2nd and 3rd line, we get

For 2nd line,

10+00+10+00+7**3+3**0+AA+75+A0+AA+7D+FA+11+1C+80+E4+7C+23+7C+4F=**79EH**

Discard carry we get, 9EH

Now 2's complement of the 9EH (1001 1110) will be (0110 0010) or 62H

But, the last byte is 02H that is not matched.

So, the information of the second line is corrupted.

For 3rd line,

07+00+20+00+DB+FE+DC+FA+DD+F6+**33 = 5DCH**

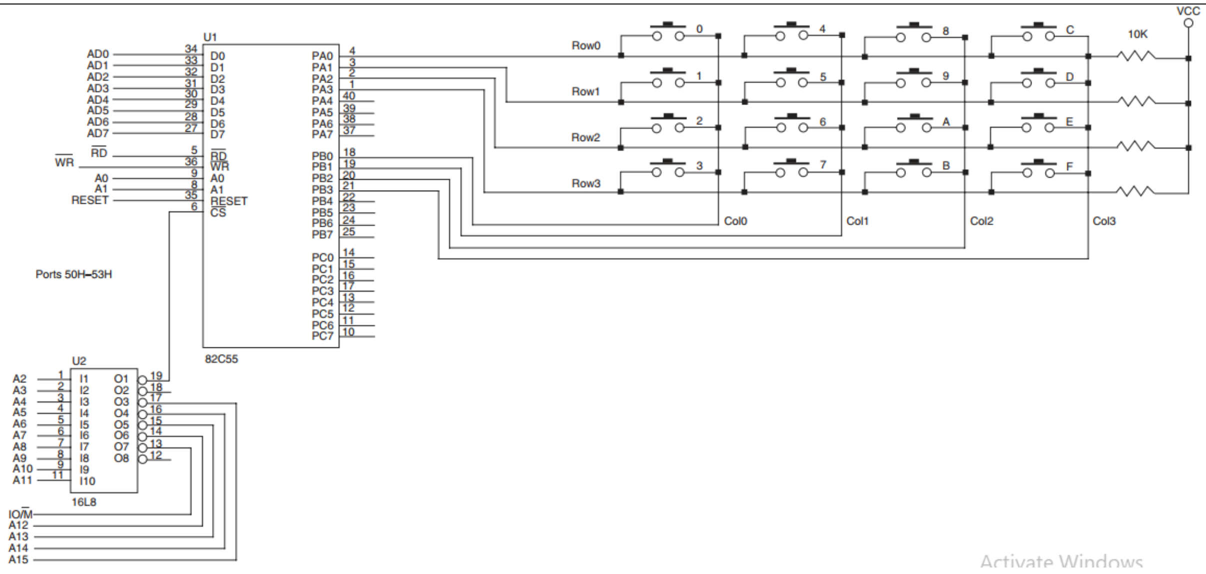
Discard carry we get, DCH

Now 2's complement of the DCH (1101 1100) will be (0010 0100) or 24H

But, the last byte is 35H that is not matched.

	So, the information of the third line is corrupted.
12	Suppose sender wants to send data 27H, 65H, 3FH, and 5AH using checksum method. Now implement the process with 8051 C program.
13	Write an 8051 Assembly program to send out a byte of data serially one bit at a time via pin P2.7. The byte comes in with the MSB first.
14	Assume that we have 4 bytes of hexadecimal data: 25H, 62H, 3FH, and 52H.(a) Find the checksum byte, (b) perform the checksum operation to ensure data integrity, and (c) if the second byte 62H has been changed to 22H, show how checksum detects the error.
15	A door sensor is connected to the P1.1 pin, and a buzzer is connected to P1.7. Write an 8051 C program to monitor the door sensor, and when it opens, sound the buzzer. You can sound the buzzer by sending a square wave of a few hundred Hz
16	The data pins of an LCD are connected to P1. The information is latched into the LCD whenever its Enable pin goes from high to low. Write an 8051 C program to send “The Earth is but One Country” to this LCD.
17	Write an 8051 C program to read the P1.0 and P1.1 bits and issue an ASCII character to P0 according to the following table. P1.1 P1.0 0 0 send ‘0’ to P0 0 1 send ‘1’ to P0 1 0 send ‘2’ to P0 1 1 send ‘3’ to P

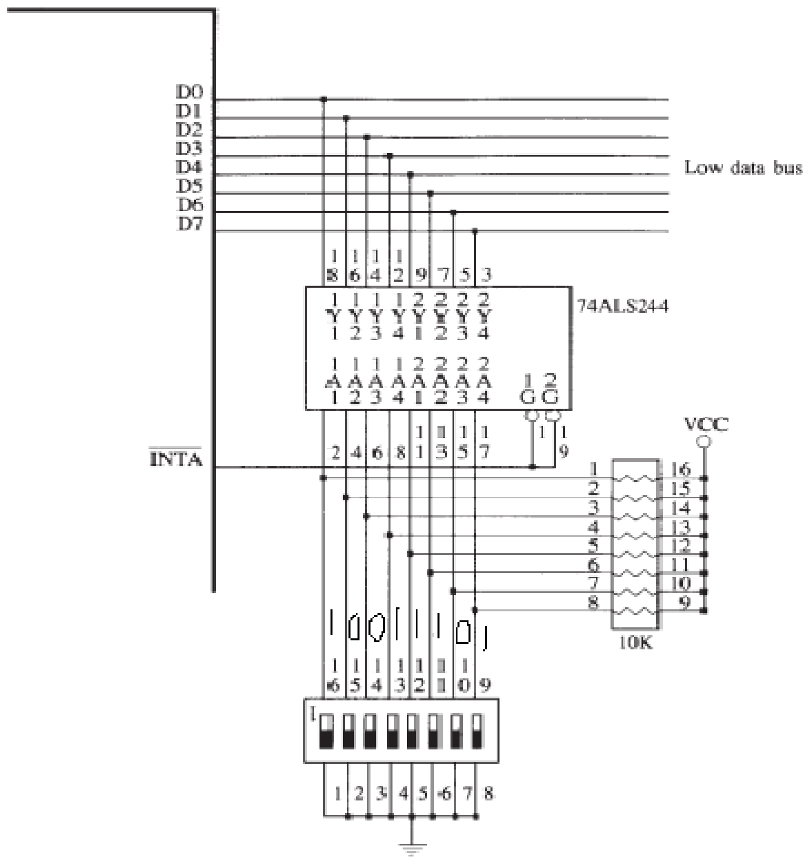
18	Write an 8051 C program to convert packed BCD 0x29 to ASCII and display the bytes on P1 and P2.
19	Write an 8051 C program to convert ASCII digits of ‘4’ and ‘7’ to packed BCD and display them on P1.
20	Write an 8051 C program to perform the checksum operation to ensure data integrity. If data is good, send ASCII character ‘G’ to P0. Otherwise send ‘B’ to P0
21	Write an 8051 C program to convert 11111101 (FD hex) to decimal and display the digits on P0, P1 and P2
22	Write a C program to send out the value 44H serially one bit at a time via P1.0. The LSB should go out first
23	Write a C program to send out the value 44H serially one bit at a time via P1.0. The MSB should go out first
24	Write a C program to bring in a byte of data serially one bit at a time via P1.0. The LSB should come in firs
25	Write a C program to bring in a byte of data serially one bit at a time via P1.0. The MSB should come in first
26	Suppose, a 4×4 key matrix given where keys are 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. You have to interface this key matrix using 82C55 with 8088 microprocessor & also explain how to detect if a key is pressed, it will be the corresponding last digit of your ID. Now **implement** this process with appropriate figure. Solution: (The Intel Microprocessor Book- Chapter11 -I/O Interface, Figure-11.25)

	 U1 AD0 AD1 AD2 AD3 AD4 AD5 AD6 AD7 RD WR A0 A1 RESET D0 D1 D2 D3 D4 D5 D6 D7 RD WR A0 A1 RESET CS PA0 PA1 PA2 PA3 PA4 PA5 PA6 PA7 PB0 PB1 PB2 PB3 PB4 PB5 PB6 PB7 PC0 PC1 PC2 PC3 PC4 PC5 PC6 PC7 82C55 Ports 50H–53H Row0 Row1 Row2 Row3 Col0 Col1 Col2 Col3 VCC 10K 0 1 2 3 4 5 6 7 8 9 A B C D E F U2 A2 A3 A4 A5 A6 A7 A8 A9 A10 A11 I1 I2 I3 I4 I5 I6 I7 I8 I9 I10 O1 O2 O3 O4 O5 O6 O7 O8 16L8 IO/M A12 A13 A14 A15 Figure: A 4 × 4 keyboard matrix connected to an 8088 microprocessor through the 82C55 PIA Port A is programmed as an input port to read the rows and port B is programmed as an output port to select a column. Here, the last digit of the ID is 3. So, we have to select column 0 keys that is 0,1,2,3. That's why 1110 is output to port B pins PB3–PB0, and column 0 will be logic 1, so the four keys in column 0 (0,1,2,3) are selected.
27	Suppose, a 4×4 key matrix given where keys are 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. You have to interface this key matrix using 82C55 with 8088 microprocessor & also explain how to detect if a key is pressed, it will be the corresponding second last digit of your ID. Now **implement** this process with appropriate figure.
28	Suppose, a 4×4 key matrix given where keys are 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. You have to interface this key matrix using 82C55 with 8088 microprocessor & also explain how to detect if a key is pressed or not. Now **implement** this process with appropriate figure.
29	You have to build a circuit that can apply any interrupt number in response to interrupt acknowledgement of 8088 microprocessor. Consider you have octal buffer 74LS244IC, DIP switches and other necessary devices. Now **illustrate** how interrupt type FFH can apply using this system with appropriate diagram.
30	You have to build a circuit that can apply any interrupt number in response to interrupt acknowledgement of 8088 microprocessor. Consider you have octal buffer 74LS244IC, DIP switches and other necessary devices. Now **illustrate** how interrupt type F5H can apply using this system with appropriate diagram.

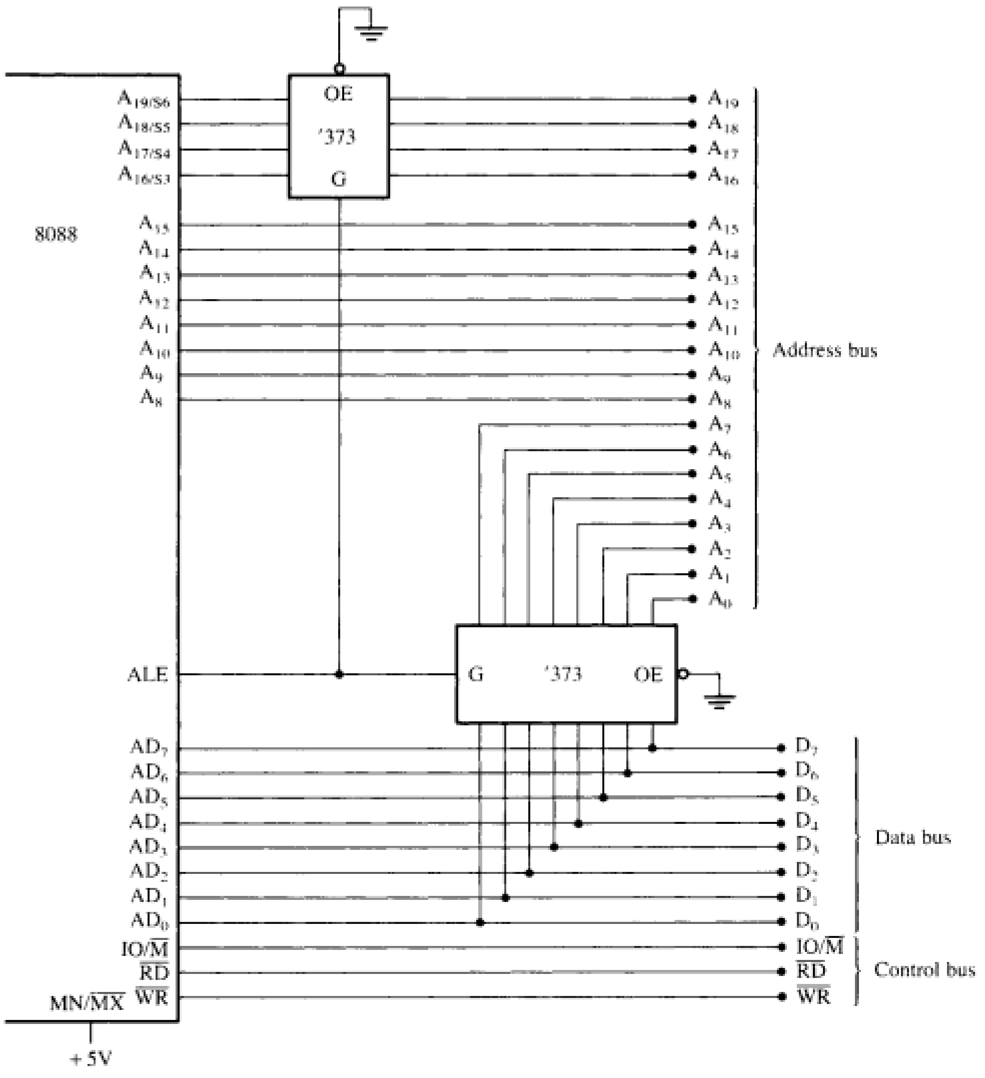
31

You have to build a circuit that can apply any interrupt number in response to interrupt acknowledgement of 8088 microprocessor. Consider you have octal buffer 74LS244IC, DIP switches and other necessary devices. Now **illustrate** how interrupt type 9DH can apply using this system with appropriate diagram.

Solution: (The Intel Microprocessor Book- Chapter12 -Interrupt, Figure-12.10)



The above figure shows how interrupt vector type number 9DH is applied to the 8088 processor data bus (D0–D7) in response to an INTR pin of 8088. In response to the INTR, the 8088 microprocessor outputs the $\overline{\text{INTA}}$ that is used to enable a 74ALS244 three-state octal buffer. The octal buffer applies the interrupt vector type number to the data bus in response to the $\overline{\text{INTA}}$ pulse. The vector type number is easily changed with the DIP switches that are shown in this illustration.

32	You have to build a circuit that can apply any interrupt number in response to interrupt acknowledgement of 8088 microprocessor. Consider you have octal buffer 74LS244IC, DIP switches and other necessary devices. Now **illustrate** how interrupt type 81H can apply using this system with appropriate diagram.
33	Before read and write operation multiplexed AD lines should be De-multiplexed of 8088/8086. Consider transparent latches74LS373 IC, 8088 IC and other necessary devices are given for this process. **Illustrate** De-multiplexing address process of 8088 based system with proper diagram. Solution:  8088 A19/S6 A18/S5 A17/S4 A16/S3 A15 A14 A13 A12 A11 A10 A9 A8 ALE AD7 AD6 AD5 AD4 AD3 AD2 AD1 AD0 IO/M̄ RD̄ WR̄ MN/MX̄ +5V OE '373 G A19 … A0 Address bus D7 … D0 Data bus IO/M̄ RD̄ WR̄ Control bus De-multiplexing process used to separate address and data bit information because any read or write operation of the processor both information are needed simultaneously.

For any read or write operation processor needs a bus cycle or 4 clock pulse time. During T1 clock pulse, ALE (address latch enable) pin of 8088 active or logic1 and address bit are available in the AD_0 pin through AD_7 pins of 8088 processor. Since, ALE is 1, the two transparent latches 74ALS373 are active and store 20-bit address information. In T2 clock pulse the ALE pin will be 0 or inactive that's why two latches are also disable, and 8-bit data information are available in the same AD_0 pin through AD_7 pins of 8088 processor as well as it stay separately from address bus. In this manner, 8088 used De-multiplexing address bus.

34 Before read and write operation multiplexed AD lines should be De-multiplexed of 8088/8086. Consider transparent latches74LS373 IC, 8086 IC and other necessary devices are given for this process. **Illustrate** De-multiplexing address process of 8086 based system with proper diagram.

Solution:

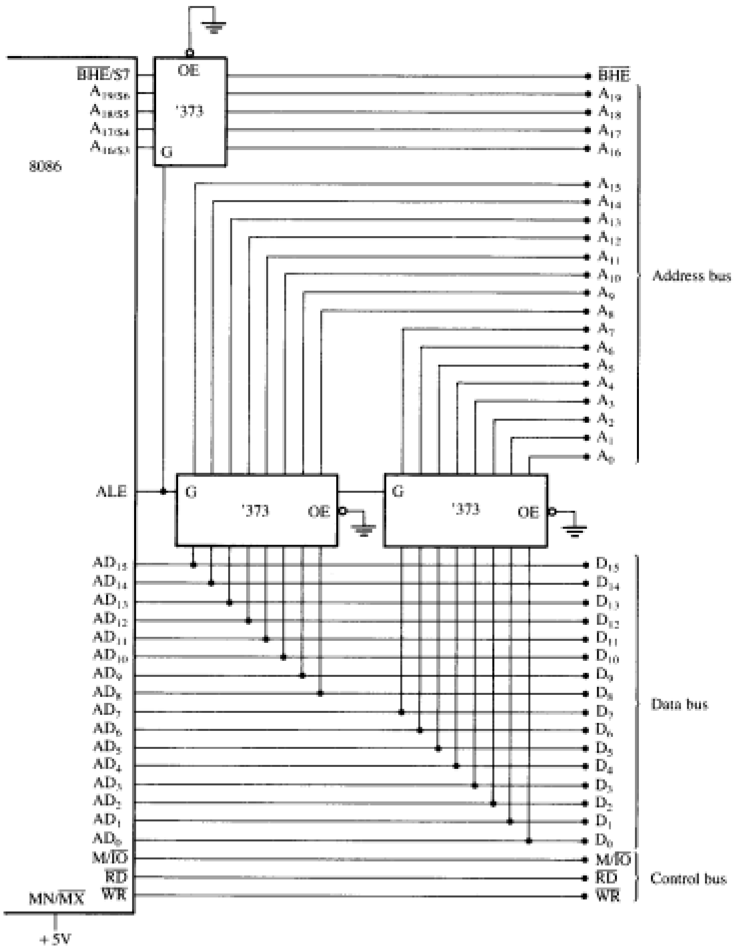


FIGURE 9–6 The 8086 microprocessor shown with a demultiplexed address bus. This is the model used to build many 8086-based systems.

De-multiplexing process used to separate address and data bit information because any read or write operation of the processor both information are needed simultaneously.

For any read or write operation processor needs a bus cycle or 4 clock pulse time. During T1 clock pulse, ALE (address latch enable) pin of 8086 active or logic1 and address bit are available in the AD_0 pin through AD_{15} pins of 8086 processor. Since, ALE is 1, the three transparent latches 74ALS373 are active and store 20-bit address information. In T2 clock pulse the ALE pin will be 0 or inactive that’s why three latches are also disable, and 16-bit data information are available in the same AD_0 pin through AD_{15} pins of 8086 processor as well as it stay separately from address bus. In this manner, 8086 used De-multiplexing address bus.

35 Suppose sender wants to send data 27H, 65H, 3FH, and 5AH using checksum method. After receiving the data receiver will perform checksum operation for data integrity. If data is not corrupted receiver will mention ‘G’ otherwise it will mention ‘B’ through port number. Now **implement** this process with 8051 C program.

Solution: (The 8051 Microcontroller Book-chapter 8051 programming in C)

Sum of 27H, 65H, 3FH, and 5AH = 125H

2’s complement of 25H (0010 0101) or checksum byte will be (1101 1011) DBH

In sender side code:

```
#include <reg51.h>

void main(void)

{

unsigned char mydata[]={0x27,0x65,0x3F,0x5A};

unsigned char sum=0;

unsigned char x;

unsigned char chksumbyte;

for (x=0;x<4;x++)

{

P2=mydata[x];

sum=sum+mydata[x];

P1=sum;

}

chksumbyte=~sum+1;
P1=chksumbyte;

}
```

In receiver side code: (Consider receiver receives all information correctly)

```
#include <reg51.h>

void main(void)

{

unsigned char mydata[] ={0x27,0x65,0x3F,0x5A,0xDB};

unsigned char shksum=0;

unsigned char x;

for (x=0;x<5;x++)

chksum=chksum+mydata[x];

if (chksum==0)

P0=‘G’;

else

P0=‘B’;

}
```

36 Suppose a door sensor is connected to the P2.3 pin, an LED is connected to P1.7, and a buzzer is connected to P3.7. Your task is to monitor the door sensor, and when it opens, sound the buzzer as well as blink the LED with 420ms delay. You can sound the buzzer by sending a square wave of a few hundred Hz. Now **implement** this task through 8051 C program.

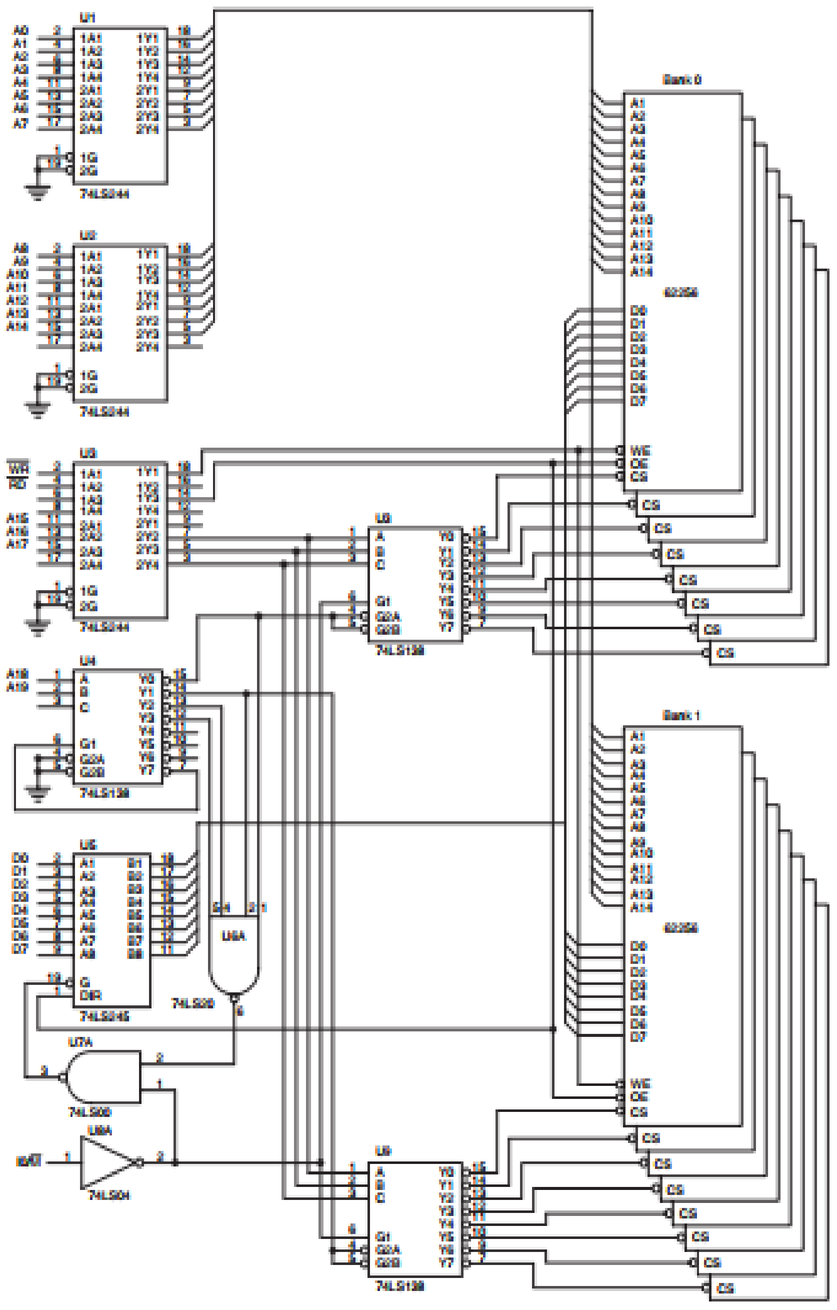
37 Suppose a door sensor is connected to the P2.1 pin, an LED is connected to P1.5, and a buzzer is connected to P2.7. Your task is to monitor the door sensor, and when it opens, sound the buzzer as well as blink the LED with 200ms delay. You can sound the buzzer by sending a square wave of a few hundred Hz. Now **implement** this task through 8051 C program.

Solution: (The 8051 Microcontroller Book-chapter 8051 programming in C)

```
#include <reg51.h>
void MSDelay(unsigned int);
sbit Dsensor=P2^1;
sbit Buzzer=P2^7;
sbit LED = P1^5;
void main(void)
{
Dsensor=1; //make P2.1 an input
while (1)
{
        while (Dsensor==1)//while it opens
        {
             Buzzer=0;
             LED=0;
             MSDelay(200);
             Buzzer=1;
             LED =1;
             MSDelay(200);
        }// end inner while loop
```

	}// end outer while loop
	
	}// end main function
	
	void MSDelay(unsigned int itime)
	{
	    unsigned int i,j;
	   for (i=0;i<itime;i++)
	       for (j=0;j<1275;j++);
	}

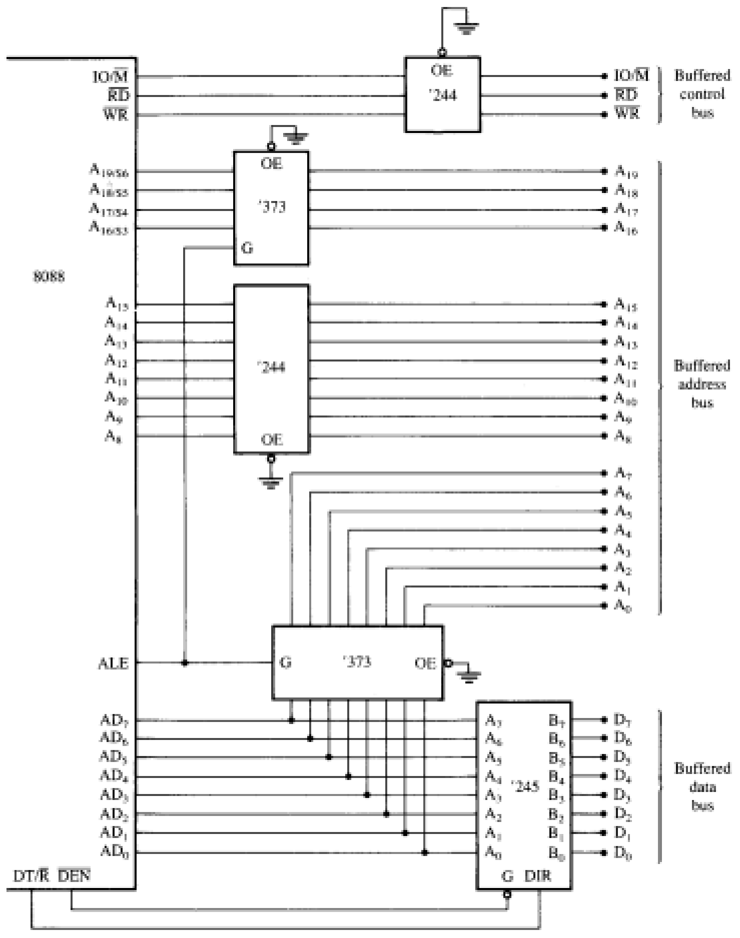
38	Suppose 768k × 8 memory portion of SRAM are separated as Bank0, Bank1, and Bank2. Your task is to build an interface system with 8088 micro-processor that read any data from Bank0 and write any data into memory location of Bank2. You have given 62256 IC which is 32K× 8 SRAM, unidirectional octal buffer 74LS244IC, bidirectional octal buffer 74LS245IC, octal decoder 74LS138IC, inverter 74LS04IC, NAND gate 74LS00IC & 74LS20IC. Now **analyze** the above system and illustrate how this task can be possible with appropriate diagram.
39	Suppose 512k × 8 memory portion of SRAM are separated as Bank0, and Bank1. Your task is to build an interface system with 8088 micro-processor that read/write any data from memory location of Bank1. You have given 62256 IC which is 32K× 8 SRAM, unidirectional octal buffer 74LS244IC, bidirectional octal buffer 74LS245IC, octal decoder 74LS138IC, inverter 74LS04IC, NAND gate 74LS00IC & 74LS20IC. Now **analyze** the above system and illustrate how this task can be possible with appropriate diagram. Solution:



In the above figure shows the interface of 512KB of SRAM with 8088 microprocessor with two separate banks namely Bank0, and Bank1. Here, bank0 address space range will be 00000H to 3FFFFH (256KB) and bank1 address space range will be 40000H to 7FFFFH (256KB). We know for read or write operation processor required a bus cycle which consists of four clock pulses.

If processor needs to write into bank1 then in a write bus cycle, for T1 processor send 20-bit physical address from the above given range like 4000FH. Here, 20-bit physical address is divided into some registers like lower15-bits (000 0000 0000 1111) goes to two 74LS244 unidirectional octal buffer and upper 3-bits (000) goes to the third 74LS244 unidirectional octal buffer which will be the same input of two decoder 74LS138 IC U3(upper side that control Bank0, eight 62256 ICs which is 32K× 8 SRAM and total space is 256KB RAM) and U9(lower

	side that control Bank1, eight 62256 ICs which is 32K× 8 SRAM and total space is 256KB RAM). Now rest of the two msb bits like A_{19} and A_{18} will send 01 to the input of the U4 decoder 74LS138IC which enable lower U9 decoder of bank1 and disable U3 decoder of bank0. In middle of the T2 clock pulse, write signal & IO/M signal is available which ensure memory interface with 8088 as well as activate write operation. That's why in the middle of T2 clock pulse 8-bit data are send from 8088 processor to 74LS245IC as bidirectional octal buffer and from this register data will send to the one of the eight enabled 62256 IC that is 32KB SRAM. Here, decoder input is (000) and it will enable first 62256IC from the eight ICs of bank1. So, 8-bit data will be ready to store the enabled SRAM IC in the corresponding 15-bits (000 0000 0000 1111) address. This write operation will need to continue for T3 clock pulse and middle of T4 clock pulse. After the middle of T4 pulse all 8-bits data will be stored in the SRAM and write signal will be inactive. In this manner write operation will be done in the Bank1 of the 512KB SRAM.
40	Suppose 1M × 8 memory portion of SRAM are separated as Bank0, Bank1, Bank2, and Bank3. Your task is to build an interface system with 8088 micro-processor that read any data from Bank1 and write any data into memory location of Bank3. You have given 62256 IC which is 32K× 8 SRAM, unidirectional octal buffer 74LS244IC, bidirectional octal buffer 74LS245IC, octal decoder 74LS138IC, inverter 74LS04IC, NAND gate 74LS00IC & 74LS20IC. Now **analyze** the above system and illustrate how this task can be possible with appropriate diagram.
41	To perform more than 10 parallel task, 8088/8086 processor is required full buffered system. Consider transparent latches74LS373 IC, 74LS244IC, 74LS245IC, 8086 IC and other necessary devices are given for this process. Illustrate fully buffered of 8086 based system with proper diagram.
42	To perform more than 10 parallel task, 8088/8086 processor is required full buffered system. Consider transparent latches74LS373 IC, 74LS244IC, 74LS245IC, 8088 IC and other necessary devices are given for this process. Illustrate fully buffered of 8088 based system with proper diagram. Solution:



Buffer system is required in 8088 to perform more than 10 task simultaneously. To synchronize the operation of 8088 it can perform 10 task simultaneously and rest of the task is queued or buffered to perform later when processor will be available. To perform fully buffer system of 8088 processor it requires transparent latches IC 74ALS373, Unidirectional octal buffer 74ALS244, bidirectional octal buffer 74ALS245ICs. In T1 clock pulse, ALE pin of 8088 will be 1 and 373 transparent latches will be enable, that's why 20-bit physical address will be buffered in two 373 transparent latches as well as one 244 octal buffer. In T2 clock pulse, ALE pin of 8088 will be 0 and 373 latches will be disable, that's why 8-bit data bit will be buffered in 74ALS245 bidirectional octal buffer. In the middle of T2 clock pulse control signals like Read or write as well as M/IO signal also buffered in 74ALS244 octal buffer. In this manner 8088 processor used buffered system.

43 Consider the following table contains the address and object file for an 8051 assembly program and, find the corresponding HEX file to load into the ROM location of 8051.

Table 1: Object file and address of an assembly program

Line Number	Address (HEX)	Object File (HEX)
01	0000	7455
02	0002	F590
03	0004	7C99
04	0006	7D67
05	0008	120012
06	000B	74AA
07	000D	F590
08	000F	120012
09	0012	F523
10	0014	80EC
11	0016	C004
12	0018	C005
13	001A	7CFF
14	001C	7DFF
15	001E	DDFE
16	001F	DCFA
17	0021	D005
18	0023	D004
19	0025	00FE

Solution:

:100000007455F5907C997D6712001274AAF59012D0

:100010000012F52380ECC004C0057CFF7DFFDDFEEF

:08002000DCFAD005D00400FE 5B

:00000001FF

44	Write a program to read the temperature and test it for the value 75. According to the test results, place the temperature value into the registers indicated by the following. If T = 75 then A = 75 If T < 75 then R1 = T If T > 75 then R2 = T
45	Write a program to transfer value 41H serially (one bit at a time) via pin P2.1. Put two highs at the start and end of the data. Send the byte LSB first
46	Write a program to bring in a byte of data serially one bit at a time via pin P2.7 and save it in register R2. The byte comes in with the LSB first
47	Assume that bit P2.2 is used to control an outdoor light and bit P2.5 a light inside a building. Show how to turn on the outside light and turn off the inside one
48	Assume that the lower three bits of P1 are connected to three switches. Write a program to send the following ASCII characters to P2 based on the status of the switches. 000 ‘0’ 001 ‘1’ 010 ‘2’ 011 ‘3’ 100 ‘4’ 101 ‘5’ 110 ‘6’ 111 ‘7’
49	(a) Write a 8051 assembly program to get hex data in the range of 00 – FFH from port 1 and convert it to decimal. Save it in R7, R6 and R5. (b) Assuming that P1 has a value of FDH for data, analyze program
50	Assume that 5 BCD data items are stored in RAM locations starting at 40H, as shown below. Write a program to find the sum of all the numbers. The result must be in BCD. 40=(71) 41=(11) 42=(65) 43=(59) 44=(37)

51	Assume that RAM locations 40 – 44H have the following values. Write a program to find the sum of the values. At the end of the program, register A should contain the low byte and R7 the high byte. 40 = (7D) 41 = (EB) 42 = (C5) 43 = (5B) 44 = (30)
52	A switch is connected to pin P1.7. Write a program to check the status of the switch and make the following decision. (a) If SW = 0, send “0” to P2 (b) If SW = 1, send “1“ to P2
53	While there are instructions such as JNC and JC to check the carry flag bit (CY), there are no such instructions for the overflow flag bit (OV). How would you write code to check OV
54	Consider you have to use bit-addressable RAM of 8051. Find out to which by each of the following bits belongs. Give the address of the RAM byte in hex (a) SETB 42H, (b) CLR 67H, (c) CLR 0FH, (d) SETB 28H, (e) CLR 12, (f) SETB 05
55	In this program, assume that the word “USA” is burned into ROM locations starting at 200H. And that the program is burned into ROM locations starting at 0. Analyze how the program works and state where “USA” is stored after this program is run.
56	Indicate which mode and which timer are selected for each of the following. (a) MOV TMOD, #01H (b) MOV TMOD, #20H (c) MOV TMOD, #12H
57	You have given 62256 IC which is 32K× 8 SRAM, 27128 IC which is 16K× 8 EPROM, dual 2 to 4 decoder 74LS139IC, three input NAND gate 74LS10IC, eight input NAND gate 74HCT30IC & 74LS240IC. Through these above devices you have to build memory ROM which size will be 64K × 8. Also, build memory RAM which size will be 128K × 8 and separated as lower & upper bank. Now your task is to build an 16-bit memory interface system with 8086 microprocessor. Also, ensure that it can read any 16-bit data from ROM as well as write 8-bit data into RAM location of upper bank. Analyze the above system and illustrate how this task can be possible with appropriate figure.

58	Suppose, 64K × 8 memory are interfaced with 8088 microprocessor. This is done using EPROM IC 2764 which size is 8K × 8 and 74LS138IC decoder. If there requires to read any data from memory location FC000H – FDFFFH, how this is possible explain with necessary diagram.
59	Assume there are 2716 IC which is 2K × 8 EPROM, 74LS133 a NAND gate decoder and 74LS04 IC. Now explain this process with necessary figure to build 8088 based memory interfacing system.
60	Find the timer's clock frequency and its period for various 8051-based system, with the crystal frequency 11.0592 MHz when C/T bit of TMOD.
61	Find the value for TMOD if we want to program timer 0 in mode 2, use 8051 XTAL for the clock source, and use instructions to start and stop the timer.
62	In the following program, we create a square wave of 50% duty cycle (with equal portions high and low) on the P1.5 bit. Timer 0 is used to generate the time delay. Analyze the program, MOV TMOD,#01 ;Timer 0, mode 1(16-bit mode) HERE: MOV TL0,#0F2H ;TL0=F2H, the low byte MOV TH0,#0FFH ;TH0=FFH, the high byte CPL P1.5 ;toggle P1.5 ACALL DELAY SJMP HERE DELAY: SETB TR0 ;start the timer 0 AGAIN: JNB TF0,AGAIN ;monitor timer flag 0 ;until it rolls over

	CLR TR0 ;stop timer 0 CLR TF0 ;clear timer 0 flag RET
63	In Execise-62, calculate the amount of time delay in the DELAY subroutine generated by the timer. Assume XTAL = 11.0592 MHz