CS/INSTR F241: Microprocessing and Interfacing

A Design Project Report on:

Door Security Control System

Submitted by :

Group 19

Sarthak Pradhan	**2013B5A8622G**
Rishabh Srivastava	**2013B5A8674G**
Sanjana Sekhar	**2013B5A7589G**
Ramchandran M	**2013B4A7541G**

Date: 23.04.16

CONTENTS

INTRODUCTION

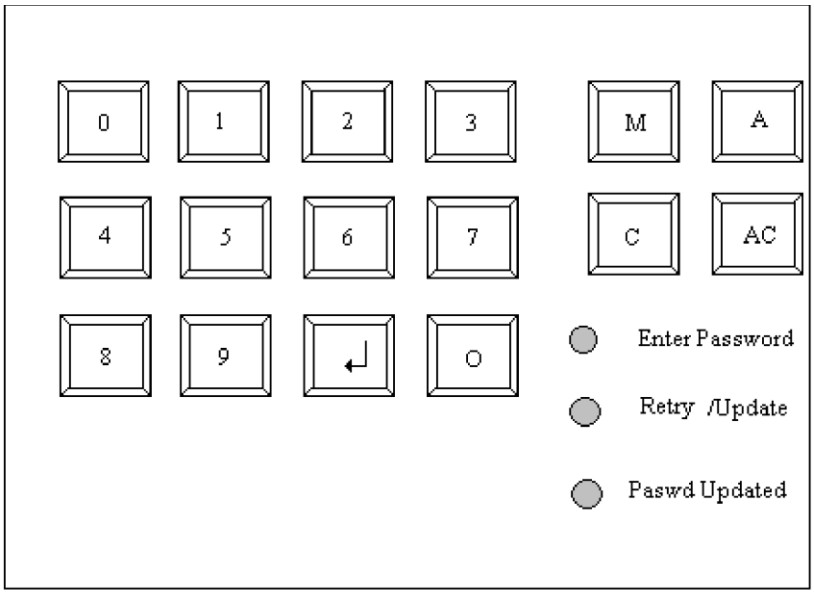
This project models a very commonly encountered system in daily life, called the Door Security System. Such systems are of utmost importance in those places where sensitive data/information is held, expensive items and equipment is housed, or important personnel are presented. To provide restricted access and protection to valuable commodities, a password controlled door is a very handy tool that can be implemented, with a basic knowledge of microprocessor, and interfacing memory and i/o devices to the microprocessor.

In this project we are implementing a door security system using the intel 8086 processor, along with timers, port controllers, and an LCD display. The interface, exact problem statement has been discussed in the coming pages. We have also shown the exact system design and specifications of hardware used. All hardware we have used is readily available in the market making this project a practical and accessible project. We hope it meets all the requirements that one might put forth.

We thank Dr. K.R Anupama and her MPI team for their impeccable documentation and thorough courseware, without which we could not have done this project. We would also like to thank our project mentors, and the instruction team for carefully and thoughtfully devising and executing the evaluation components, all of which helped strengthen our concepts and take this design to completion. This project has been an enjoying and enriching experience for our team.

PROBLEM STATEMENT

Description: This system controls the opening and closing of a door based on password entry. If the password is correct the person can enter. Each person is given two chances to enter the correct password. On failure an alarm is sounded. Inside the room a button is available when the button is pressed the door opens for I Min, so that the person can leave the room.



User Interface: There are three set of passwords: (1) User (2) Master (3) Alarm off

- The Master password is used by the security Personnel for updating Password of the day. Pressing the M button activates this mode. The system glows Enter Password LED asking the personnel to enter the password. The master password is a 16-digit value. The master is given only a single chance to enter the password. If authenticated, the retry/Update LED glows. If there is a failure in authentication the alarm is sounded. When the retry/ Update LED glows the user has to enter password of the day. This is 12-digit value. Once this value has been accepted by the system the Passwd Updated LED glows.
- User has to press the O key when he wants to enter the room. The Enter Password LED prompts the user to enter the password. The user is given C/AC option as well. If the first attempt fails, the RETRY LED glows. The user is allowed to re-enter password, on authentication door opens for a period of 1 Min. On Failure an ALARM is sounded.
- To Turn-off the Alarm the A button has to be pressed. Enter Password LED glows prompting user to enter the 14-digit password for turning of alarm, no retries are allowed. If authentication is successful then the alarm is turned off.
- To leave the room a button is available inside the room, when the button is pressed the door opens for 1 Minute so that the person can leave the room.

ASSUMPTIONS

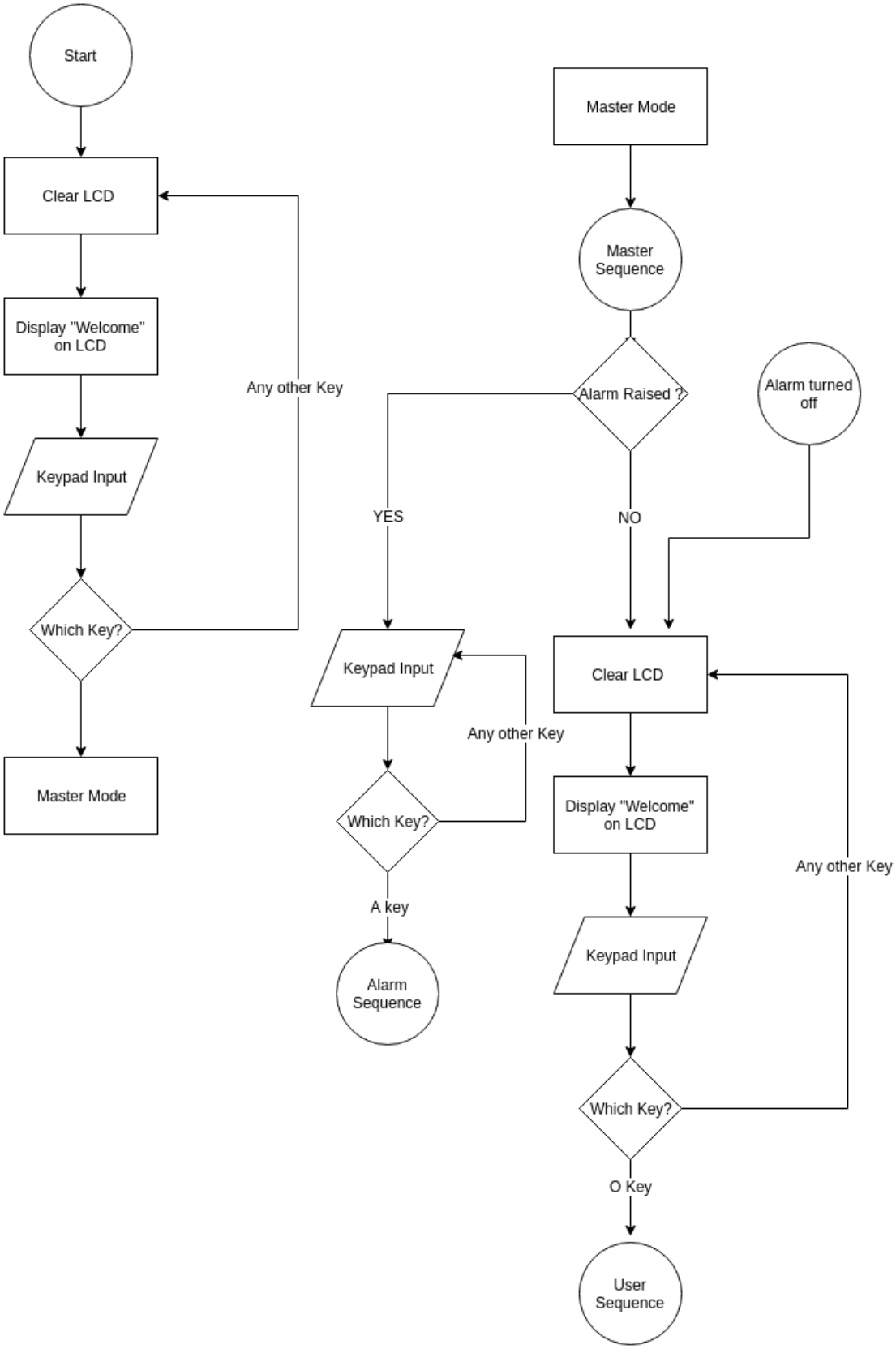
1. 1. Only one person may open and pass the door at a time, i.e. one User mode execution implies only 1 person opening the door and going inside.
2. All operations are sequential, no two operations will happen simultaneously and require greater attention from 8086 at the same instant of time, and hence there is no need for a priority resolver such as the interrupt controller 8259.
3. User will not try to open the door from inside or outside, at the 24 hour mark after Master has set password of the day. Interrupt for 24 hour completion and 1 minute door close will NOT occur at the same time.
4. Alarm can only be turned off from outside the door.
5. Once a particular mode has been entered, User/Master must complete the procedure, i.e. M or O or A pressed within execution of a particular mode will have no effect. (This part has been ensured in the code.)
6. If a mistake is made in entering password for switching the alarm off, then system will enter a lock-down state till power is cut.
7. The first time the 8086 is switched on, the 24 hour clock starts running from that instant, and there onwards computes 24 hours count. Therefore the first action to be performed is to set the Password of the Day by Master in Master Mode.
8. As soon as the machine is switched on the user is supposed to enter Master Password, hardcoded as:

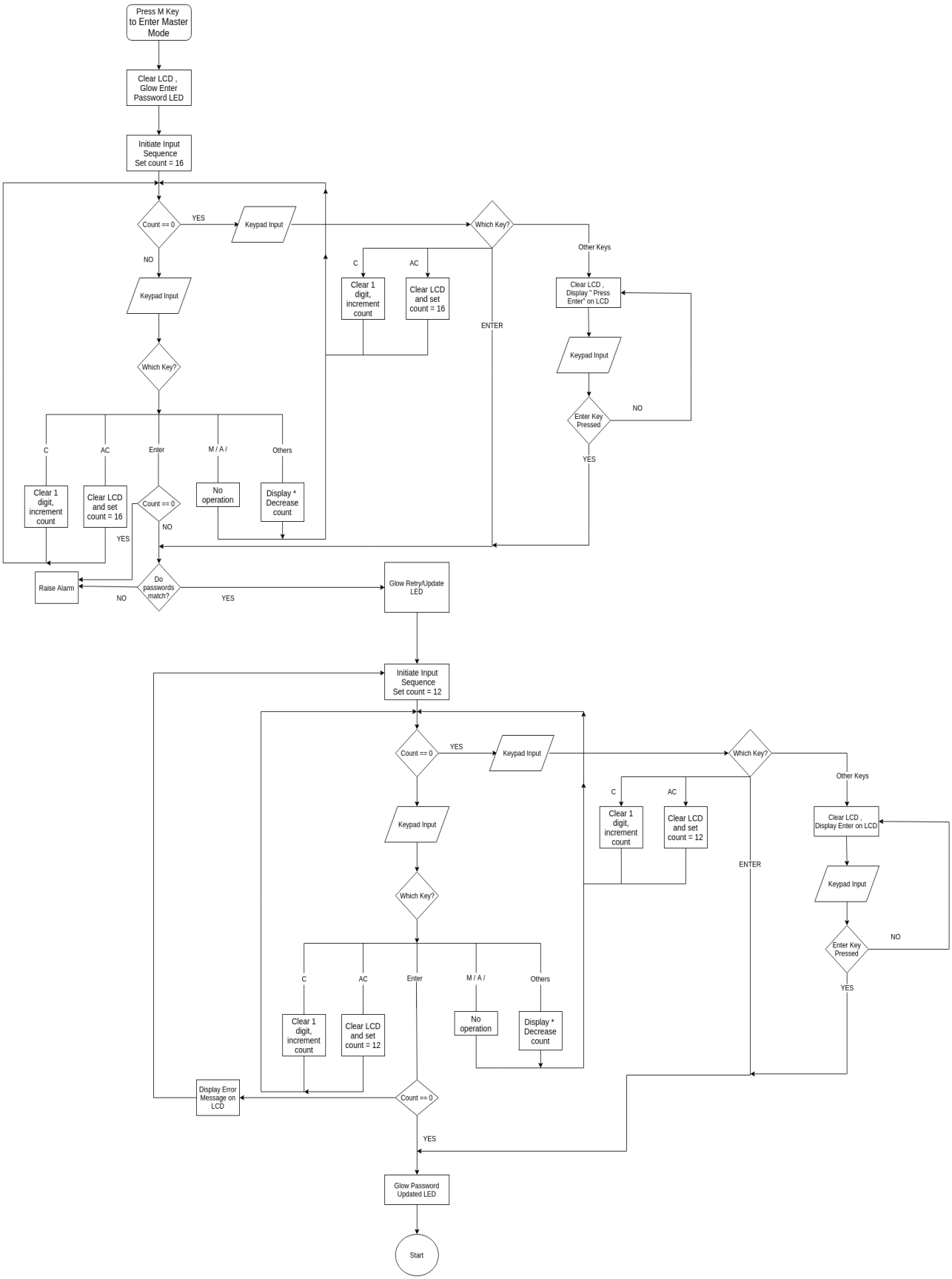
   MASTER MODE PASSWORD : 9763106626976310

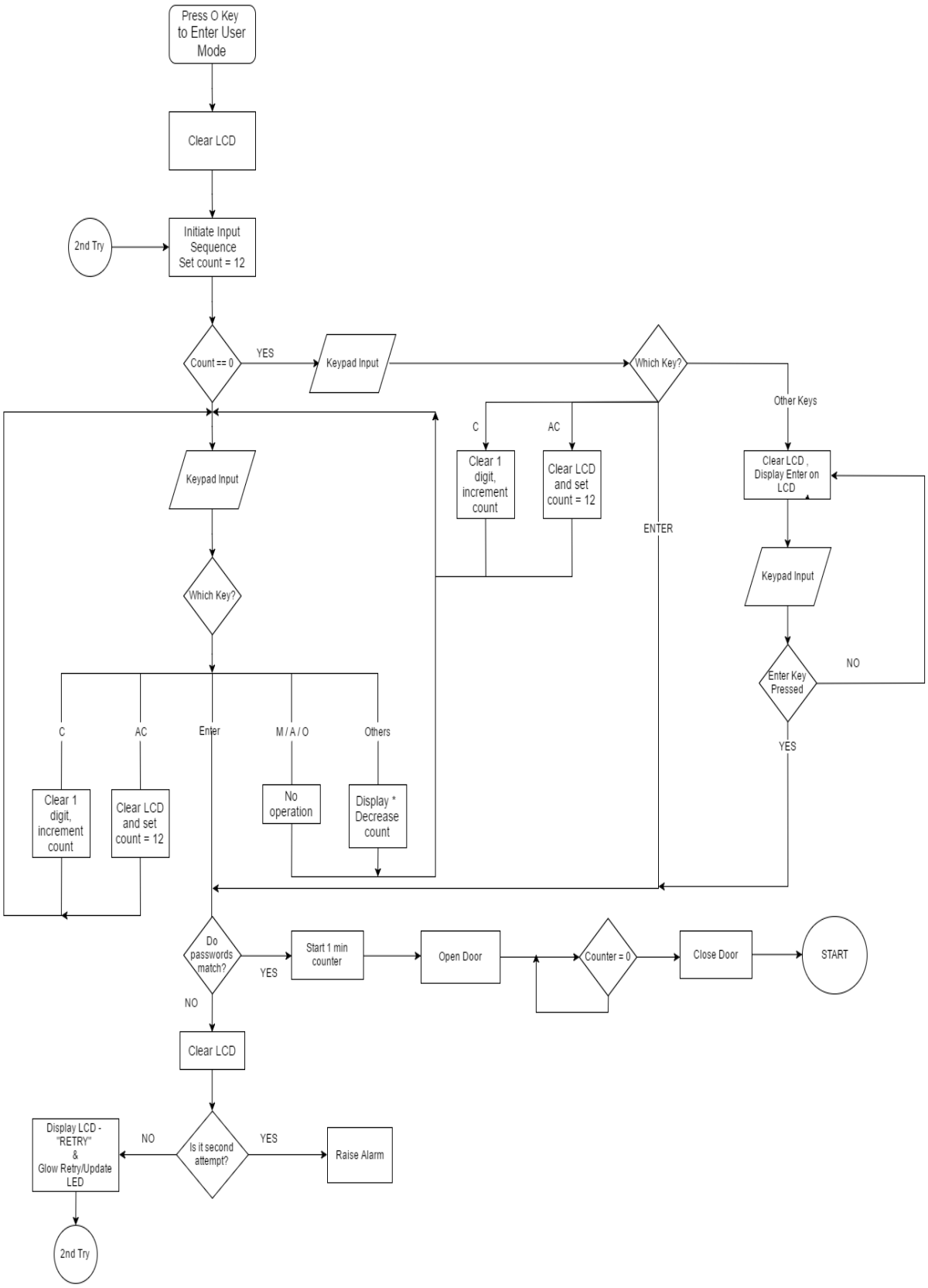
   The alarm password is also hardcoded as:

   ALARM RESET PASSWORD : 99999999999999

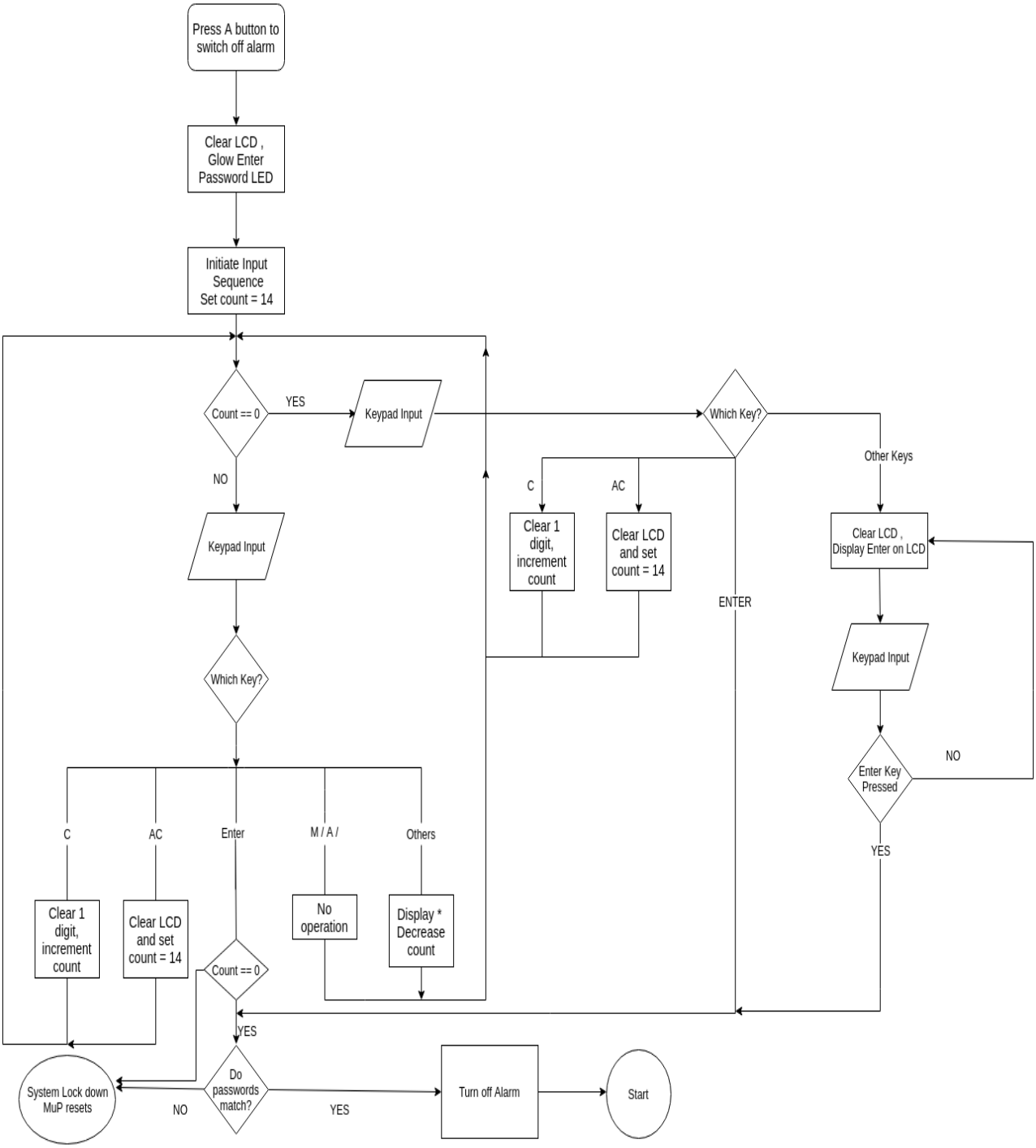
FLOWCHART


Start
Clear LCD
Display "Welcome" on LCD
Keypad Input
Which Key?
Any other Key
Master Mode
Master Mode
Master Sequence
Alarm Raised ?
Alarm turned off
YES
NO
Keypad Input
Clear LCD
Any other Key
Which Key?
Display "Welcome" on LCD
A key
Alarm Sequence
Keypad Input
Any other Key
Which Key?
O Key
User Sequence


Press M Key to Enter Master Mode
Clear LCD , Glow Enter Password LED
Initiate Input Sequence Set count = 16
Count == 0
YES
Keypad Input
Which Key?
NO
Keypad Input
Which Key?
C
AC
Clear 1 digit, increment count
Clear LCD and set count = 16
Other Keys
Clear LCD , Display " Press Enter" on LCD
ENTER
Keypad Input
Enter Key Pressed
NO
YES
C
AC
Enter
M / A /
Others
Clear 1 digit, increment count
Clear LCD and set count = 16
Count == 0
NO
YES
No operation
Display * Decrease count
Raise Alarm
Do passwords match?
NO
YES
Glow Retry/Update LED
Initiate Input Sequence Set count = 12
Count == 0
YES
Keypad Input
Which Key?
Keypad Input
C
AC
Clear 1 digit, increment count
Clear LCD and set count = 12
Other Keys
Clear LCD , Display Enter on LCD
ENTER
Keypad Input
Enter Key Pressed
NO
YES
C
AC
Enter
M / A /
Others
Clear 1 digit, increment count
Clear LCD and set count = 12
No operation
Display * Decrease count
Display Error Message on LCD
Count == 0
YES
Glow Password Updated LED
Start


Press O Key to Enter User Mode
Clear LCD
2nd Try
Initiate Input Sequence Set count = 12
Count == 0
YES
Keypad Input
Which Key?
Other Keys
C
AC
Clear 1 digit, increment count
Clear LCD and set count = 12
ENTER
Clear LCD , Display Enter on LCD
Keypad Input
Enter Key Pressed
NO
YES
Keypad Input
Which Key?
C
AC
Enter
M / A / O
Others
Clear 1 digit, increment count
Clear LCD and set count = 12
No operation
Display * Decrease count
Do passwords match?
YES
Start 1 min counter
Open Door
Counter = 0
Close Door
START
NO
Clear LCD
Display LCD - "RETRY" & Glow Retry/Update LED
NO
Is it second attempt?
YES
Raise Alarm
2nd Try

Text


Press A button to switch off alarm
Clear LCD , Glow Enter Password LED
Initiate Input Sequence Set count = 14
Count == 0
YES
Keypad Input
NO
Keypad Input
Which Key?
C
AC
Enter
M / A /
Others
Clear 1 digit, increment count
Clear LCD and set count = 14
No operation
Display * Decrease count
Count == 0
YES
System Lock down MuP resets
NO
Do passwords match?
YES
Turn off Alarm
Start
Which Key?
C
AC
Other Keys
Clear 1 digit, increment count
Clear LCD and set count = 14
ENTER
Clear LCD , Display Enter on LCD
Keypad Input
Enter Key Pressed
NO
YES

DESIGN SPECIFICATIONS

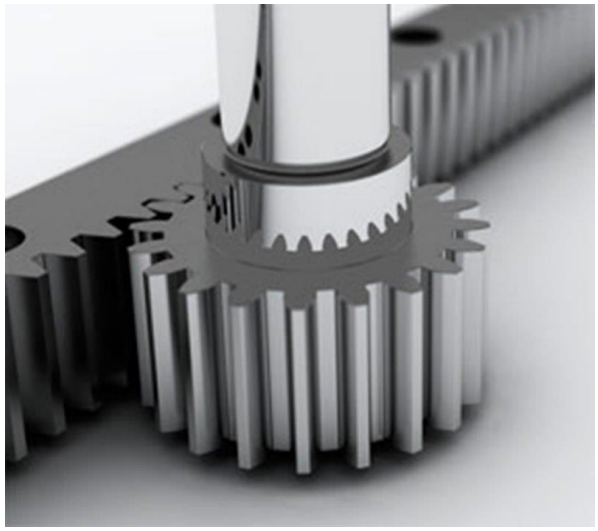
Hardware Used :

S.No.	Hardware Device	Description	Quantity
1	8086	16 Bit Microprocessor(Intel)	1
2	74LS373	Octal Latch	3
3	8284	Clock Generator	1
4	74LS245	Transceiver	2
5	74155	2-4 Decoder	1
6	7432	OR Gates	6
7	2716	ROM Chips	4
8	6116	Static RAM Chips	2
9	74138	3-8 Decoder	1
10	8253	Programmable Interval Timer	2
11	8255	Programmable Peripheral Interface with I/O	2
12	N/A	Hex-Keypad	1
13	HD77420	LCD Driver (Hitachi)	1
14	LM016l	LCD Display	1
15	PSM 57-81 2P	Stepper Motor(MAXIM)	1
16	C503B	LEDs (RED)	3
17	C503B	LEDs (BLUE)	1
18	ULN2003A	Darlington Transistor	1
19	N/A	Push Button/Switch	1
20	74LS241	Octal Tri-state Buffer	1
21		ALARM(Siren)	1

Door Specifications:

Door Material : Glass (Density = 2595 kg/cm^3)
Height = 78 inches(1.9812m)
Width = 32 inches(0.8128m)
Thickness = 0.5 inches(0.0127m)
Mass = 117 lbs (53 Kgs)

We will be using door sliding mechanism. The door's base is attached to sliding rails. This is shown in the figure below. The motor will rotate and the door can slide in two directions through a gear mechanism. A gear will be attached to the motor through a beam coupling. Then through a train of gears the final gear will be in contact with the door.



Calculations:

MOTOR TO DOOR INTERFACING:

We will be using Maxima Stepper Motor PSM 57-81 2P (Holding Torque 20.4 Kg-cm) to control the motion of the door. The sliding rails are made of Aluminium and are well lubricated. The coefficient of friction is assumed to be 0.3. The weight of the door is 53kgs. Therefore the force required to slide the door is 0.3 x 53 kg force = 15.9 kg force. The holding torque of motor is 20.4 Kg-cm.

The gear attached to the motor through a beam coupling is of radius 2 cm. Therefore Radius of second gear which is connected to the door should be = (15.9/10.2)*2 = 3.11 cm. We have assumed Radius ratio = Gear ratio.

Step angle of the motor = 1.8 degrees. Therefore it needs 100 steps to complete 1 revolution. Now the width of the door = 0.8128 m. Therefore we consider that gear teeth are attached to 0.8 m width of the door symmetric about the centre. Radius of gear attached to the motor = 2 cm. Therefore in one revolution of motor 2 x 3.1415 x 2 cm = 12.566 cm is covered. It has to cover 81.28 cms. Therefore no of revolutions of motor required = 81.28/12.566 = 6.46 revolutions. Therefore it has to take 6.46 x 100 steps to complete that. That is it has to take 646 steps.

Motor Specifications :

Model Number	PSM 57-81 2P
Current	4A
Resistance(ohms)	0.4
Inductance(mH)	1.8
Dentent Torque(g/cm)	700
Rotor Inertia(g/cm^2)	480
Weight (Kg)	1.1
Motor Length(mm)	81
Shaft Diameter(mm)	8
Holding Torque(Kgcm)	0.8 x 15

Counter Calculations **:**

8253-1 counter 0 is fed with a 2.5Mhz frequency signal from 8284 p-clock.This is reduced further to 50Hz.

Out clock freq = In clock freq/count

Count = **50,000**

This 50Hz clock signal is fed to 8253-1 counter 1 which is also configured in Mode 3 (Rate Generator).50Hz is further reduced to 0.5Hz by loading a count value of

50/0.5=**100**

24 hour Timer : This timer Operation is implemented by 8253-2 counter 0.
(Mode 2,Rate generator or Divide by N counter)

The clock to this counter is fed from the output of 8253-1 counter 1 which is configured in Mode 3(Square Wave Generator).The clock frequency applied to 8253-2 counter zero is 0.5Hz.

0.5*60*60*24 = **43200**

Thus by loading a count of 43200 in this counter a 24 hour timer is implemented. The out pin of this counter is inverted and fed to NMI pin of 8086. At the end of 24 hours a LOW-to-HIGH pulse is generated at NMI which triggers INT 02h.In response, 8086 branches to ISR meant for INT 02h(Nmi_24hrtimer) and activates the Master Mode Key(M) on the Keypad. “UPDATE DAY PASS” is flashed on the LCD module and the code dosen’t branch back until a new password for the user mode is set. On setting the new User password the system branches back to normal routine and the user is greeted with LCD flashing “WELCOME”.

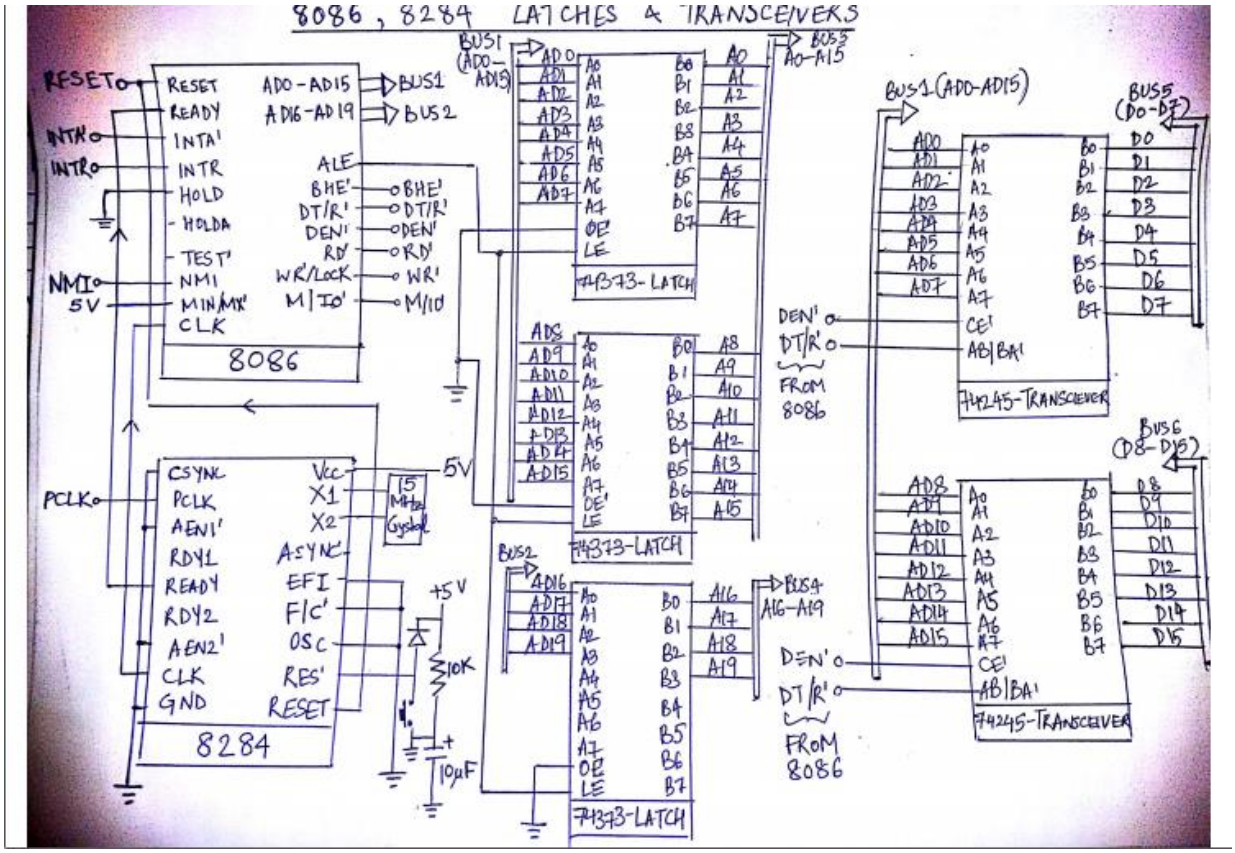
1 minute Timer : This timer operation is implemented using 8253-1 counter 2.
This counter is fed with a clock frequency of 50Hz from 8253-1 counter 1 out pin.8253-1 counter 2 is configured in Mode 1(h/w re-Triggerable one shot timer).

50*60 = **3000**

Thus by loading a count of 3000 in 8253-1 counter 2 a 1 min timer is implemented. As soon as the open_door subroutine is called the timer is triggered by giving a LOW-HIGH-LOW pulse on its gate pin.On receiving this signal the out pin goes low(Logic 0).At the end of one minute,(count == 0) out pin goes high(Logic 1) and this out pin is polled at PC1 pin of 8255-2.At the end of 1 minute close_door subroutine is called and door closes.

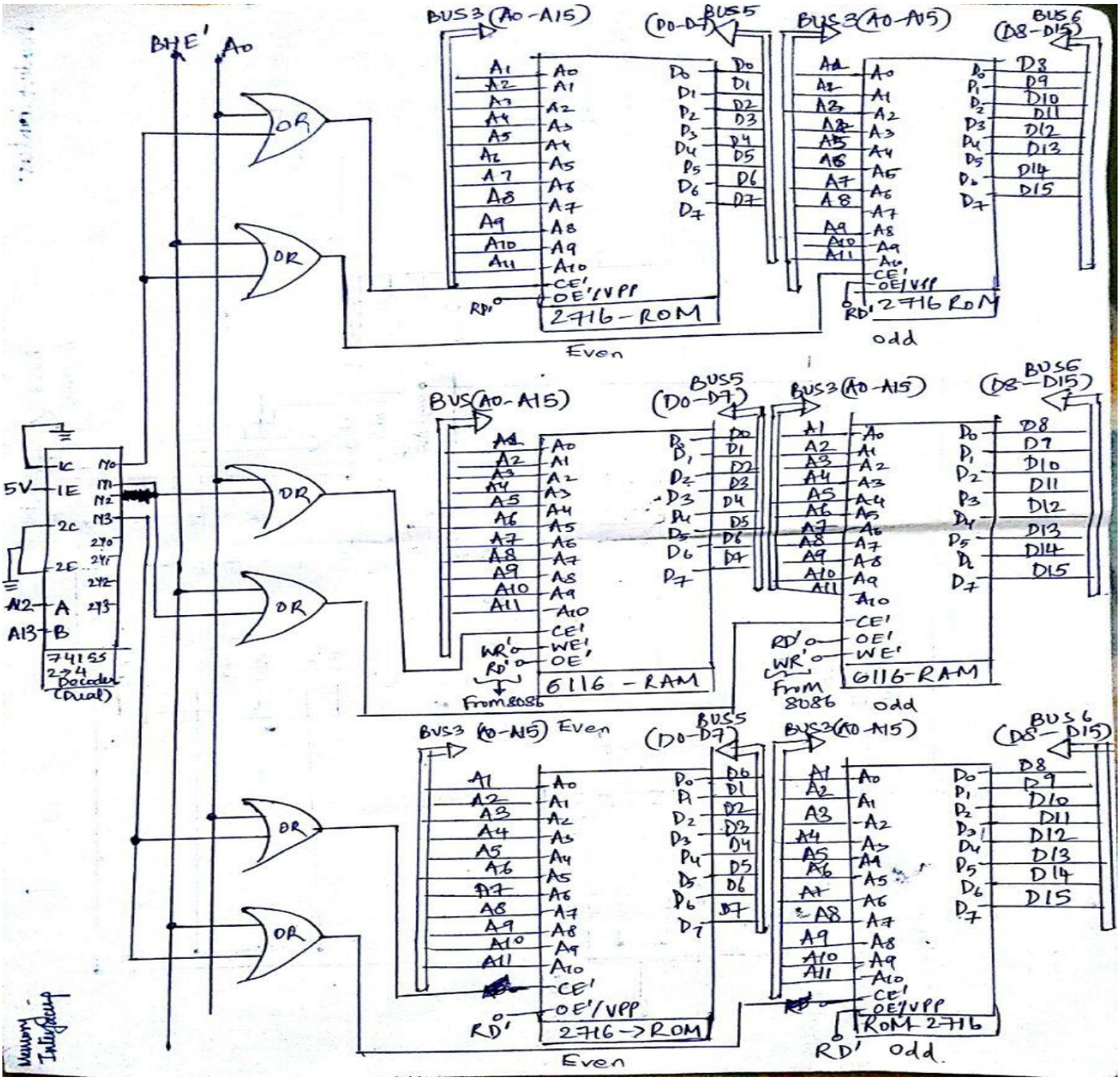
CIRCUIT DIAGRAM

6.1) 8086, 8284, Latches and Transceivers

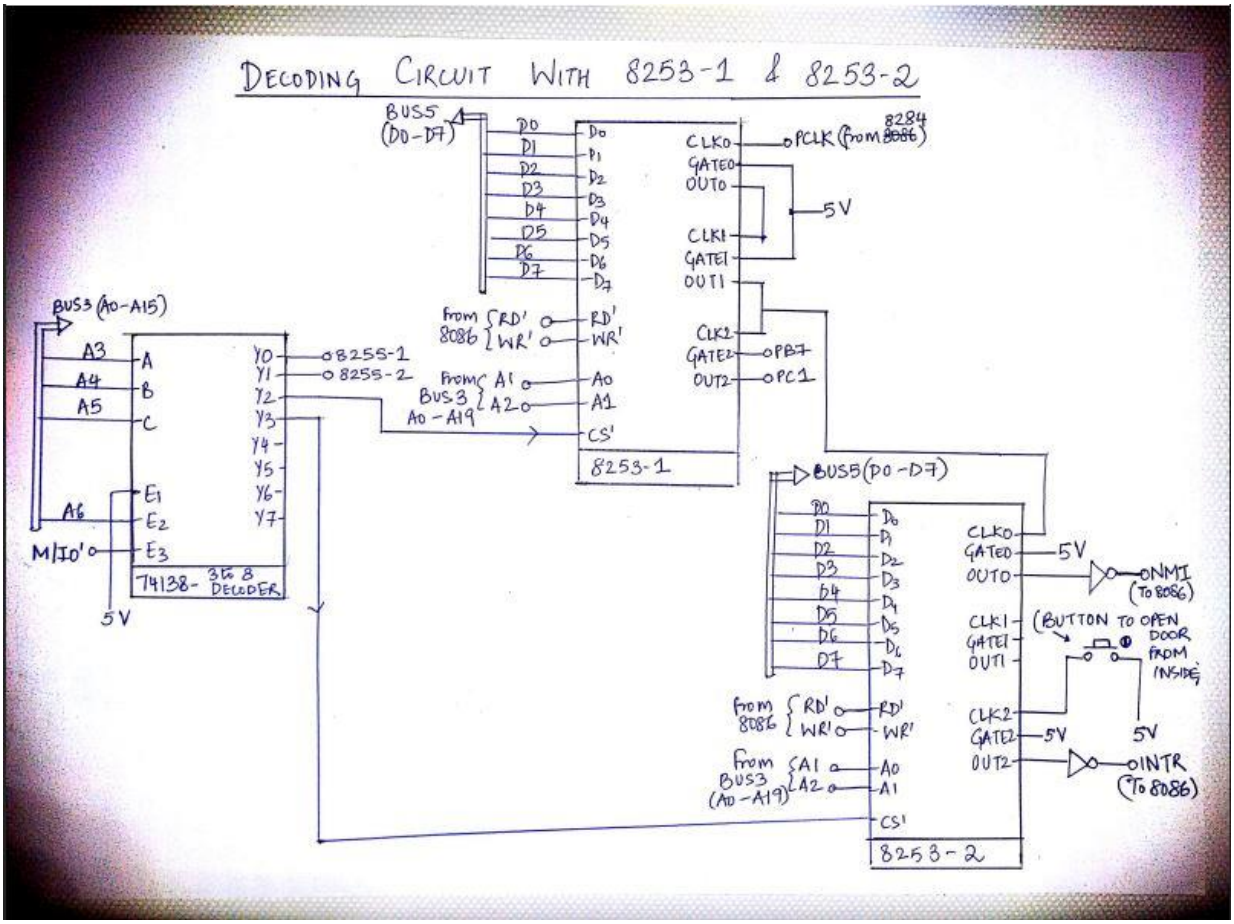


6.2) Memory Interfacing:

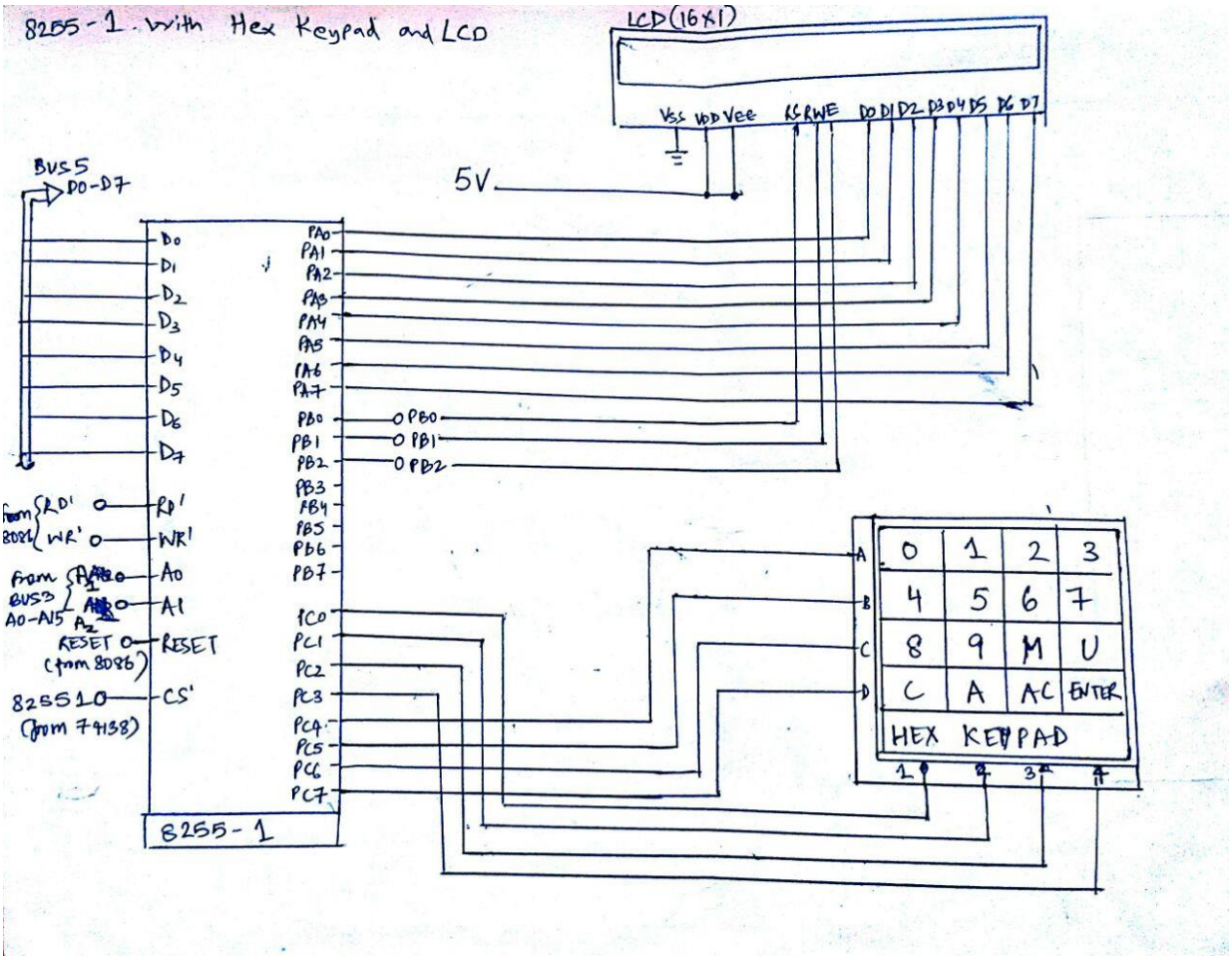
ROM-4k(2k-even,2k-odd)	00000h-00FFFh
RAM-4k(2k-even,2k-odd)	01000h-01FFFh
ROM-4K(2k-even,2k-odd)	FF000h-FFFFFh


BHE'
A0
OR
OR
OR
OR
OR
5V
A12
A13
74155
2-4
Decoder
(Dual)
BUS3 (A0-A15)
BUS5 (D0-D7)
BUS6 (D8-D15)
2716-ROM
Even
2716 ROM
odd
BUS(A0-A15)
6116 - RAM
From 8086
WR'
RD'
WE'
OE'
CE'
OE'/VPP
6116-RAM
2716->ROM
RD' odd
Even

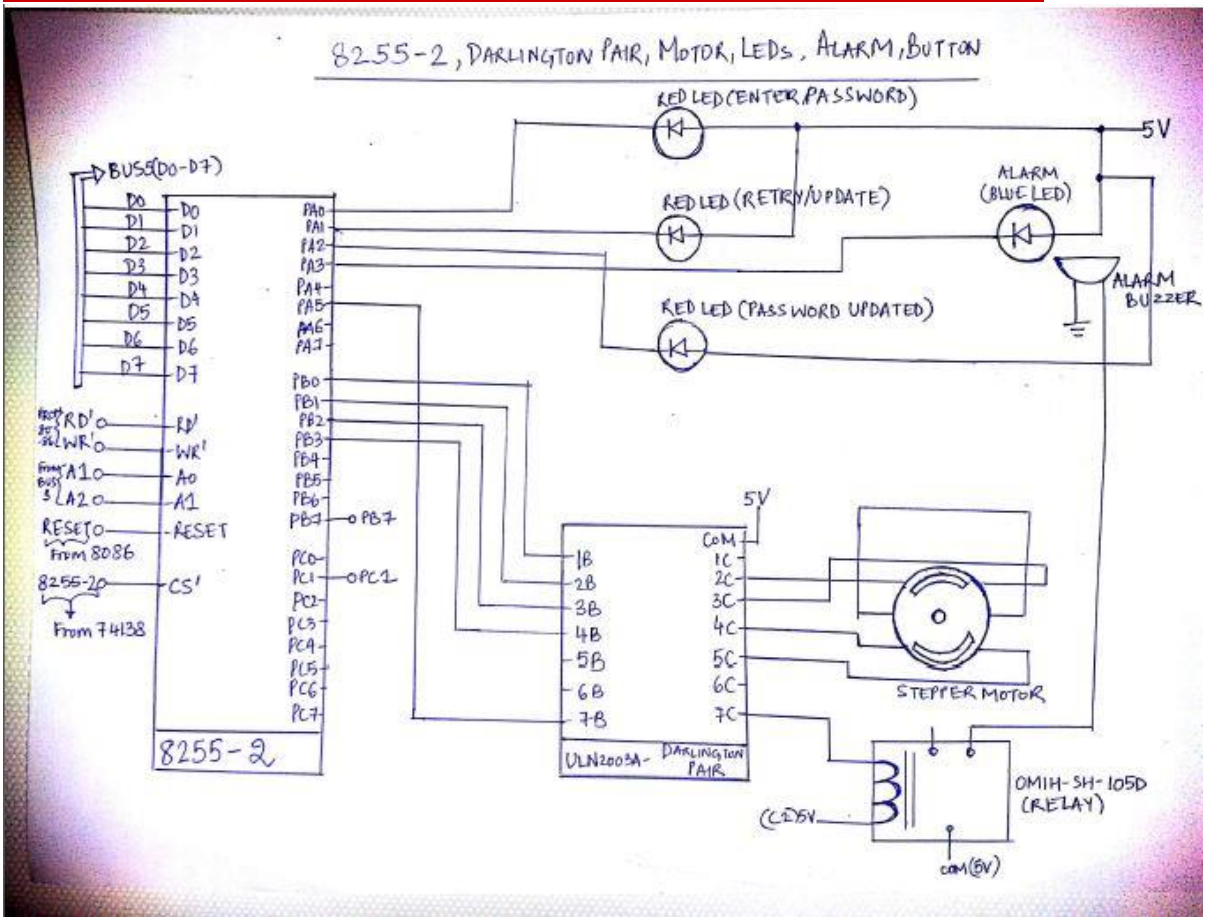
6.3) Decoding Circuit with 8253-1 and 8253-2



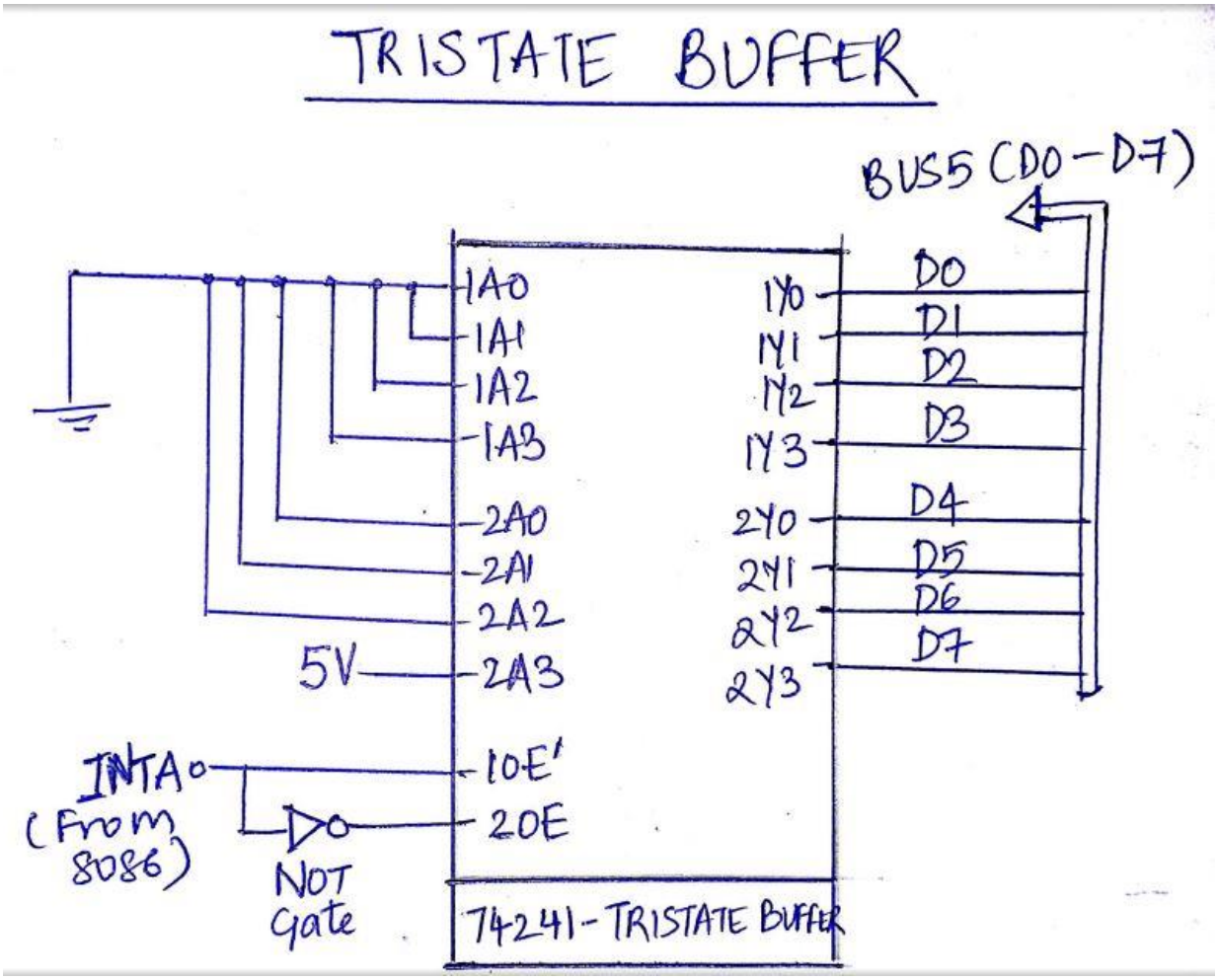
6.4) 8255-1 with Hex Keypad and LCD



6.5) 8255-2 with Darlington pair, Motor, LEDs, Alarm, Inside Button



6.6) Tristate Buffer for Interrupt



CODE DOCUMENTATION

Here is a brief description of the Subroutines. They appear in the order of usage in the source code.

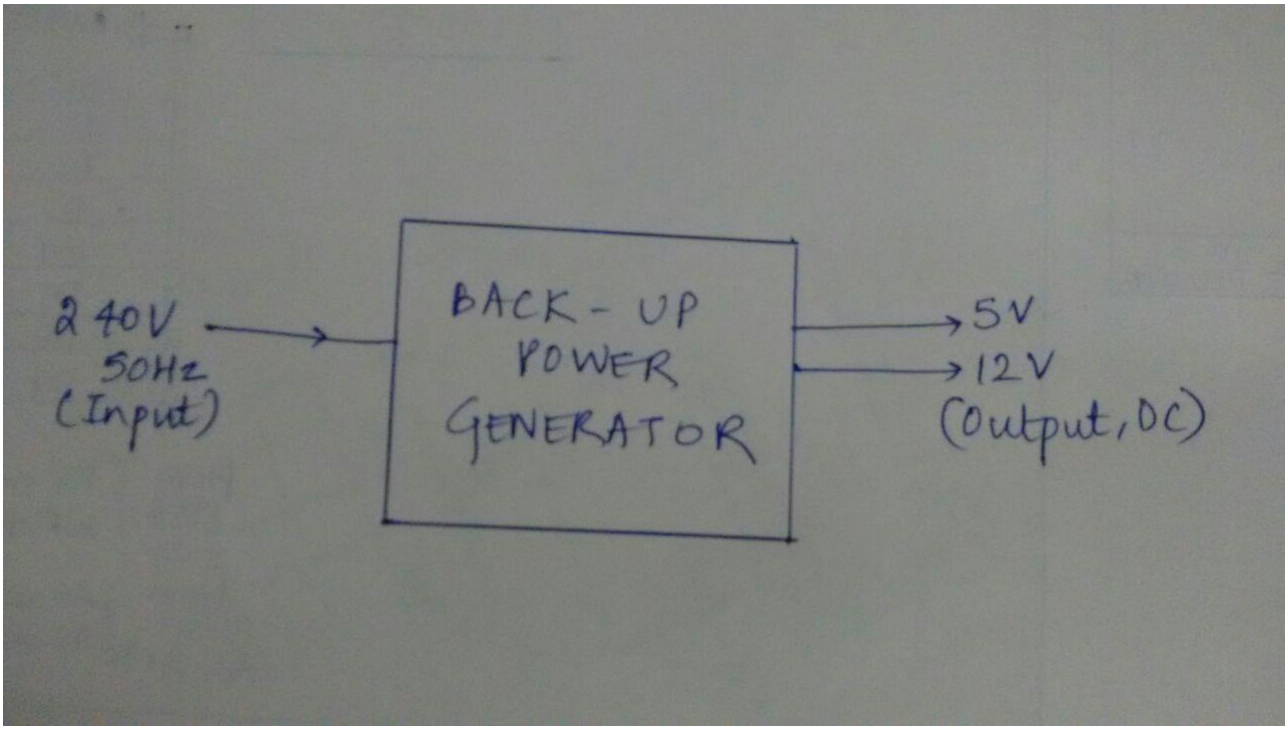
S.No	Name	Description
1	clear_LCD	Clears the LCD Display
2	welcome_msg	Prints “ Welcome“
3	keypad_input	Obtains key pressed from the hex-keypad input
4	intm	Initiates the Master Sequence
5	inta	Initiates the Alarm Sequence
6	intu	Initiates the User Sequence
7	DELAY_20ms	20 milliseconds delay
8	DELAY_0.04s	0.04 seconds delay
9	DELAY_max	Max time delay possible
10	enter_LCD	Prints “ PRESS ENTER” on LCD
11	Print_*	Prints “*” on LCD
12	ints	Initiates the Switch Sequence
13	open_door	Rotate motor 90 deg to open it
14	close_door	Makes the motor rotate in opposite direction
15	update_msg	Prints “UPDATE PASSWORD” on LCD
16	clear_1digit_LCD	Shifts pointer left by 1 , prints Space , Shifts pointer again left by 1
17	error_msg	Prints “ ENTER 12 DIGITS” on LCD
18	retry_msg	Prints “ RETRY” on LCD
19	updateday_msg	Prints “UPDATE DAY PASS” on LCD

INTERRUPTS

In the code we have used two interrupts(NMI and INT 80h). NMI is used for implementation of the 24 hour timer which is utilised to set the password of the day. INT 80h is fired when the push-button is pressed. It is pressed to open the door from inside. INTR signal of 8086 happens to be level triggered, so for one time firing of this interrupt the push button is connected to the clock pin of 8253-2 counter 2. 8253-2 counter 2 is configured as Mode 2.The out pin is connected to INTR pin of 8086.The interrupt vector is generated via a Octal Tri-state buffer. Octal Tri-state buffers puts 80h on the data lines (D0,D7) as soon as INTA pin of the 8086 goes low which enables the Octal Tri-state buffer(74LS241).

What happens in case of power failure ?

The main power supply has battery backup system. As we didn't have any secondary memory, the memory will be lost in case of full battery drain. However, when there is a secondary memory interfaced, a backup can be created before power drain and system will restore full operation when power is restored.



Stepper Motor :

The command to open the door is issued from 8255-2 port when the routine open_door is called in the main routine. Appropriate pins of ULN2003A Darlington pair are pulled high which in turn drive the motor.

ALP CODE

```
        jmp     st1
       db     5 dup(0)
       ;IVT entry for NMI (INT 02h)
       dw      Nmi_24hrtimer
       dw     0000
       db     500 dup(0)
                        ;IVT entry for 80H
                        dw     Switch_intR
       dw     0000
                        db     508 dup(0)
                       st1:     cli


                        ;intialize   ds,   es,ss   to
start of RAM
        mov      ax,0200h
        mov      ds,ax
        mov      es,ax
        mov      ss,ax
        mov      sp,0FFFEH
;  INITIALIZATION  OF  8255,8253  BEGINS
HERE


            sti
   mov al,89h  ; control word for 8255-2
            out 0Eh,al


            mov al,88h              ;       control
word for 8255-1
            out 06h,al


            mov al,36h              ;control
word for 8253-1 counter 0
            out 16h,al


            mov   al,56h     ;control   word   for
8253-1 counter 1
            out 16h,al


            mov   al,92h     ;control   word   for
8253-1 counter 2
            out 16h,al


            mov   al,34h     ;control   word   for
8253-2 counter 0
            out 1eh,al


            mov   al,5ah     ;control   word   for
8253-2 counter 1
            out 1eh,al


            mov   al,94h     ;control   word   for
8253-2 counter 2
            out 1eh,al


            mov al,50h              ;load   count
lsb for 8253-1 counter 0
            out 10h,al


            mov  al,0C3h ;load  count  msb  for
8253-1 counter 0
            out 10h,al


            mov al,64h              ;load   count
for 8253-1 counter 1
```

```
        out 12h,al

        mov al,5h ;load count lsb for
8253-1 counter 2 (1 minute Timer)
        out 14h,al



        mov al,10 ;load count for 8253-2
LSB counter 0    (24 hour counter)
        out 18h,al

        mov al,0  ;load count for 8253-2
MSB counter 0    (24 hour counter)
        out 18h,al

        mov al,3  ;load count for 8253-2
counter 1    (Switch trigger counter)
        out 1ah,al

        mov al,2  ;load count for 8253-2
counter 2
        out 1ch,al
;INITIALIZATION OF 8255,8253 ENDS HERE



        mov al,00h              ;default low
output from 8255-2 upper port C
        out 0ch,al

        call      DELAY_20ms      ;LCD
INITIALIZATION BEGINS
        mov al,04h
        out 02h,al
        call DELAY_20ms
        mov al,00h
        out 02h,al

        mov al,38h
        out 00h,al

        mov al,04h
        out 02h,al
        call DELAY_20ms
        mov al,00h
        out 02h,al
        call DELAY_20ms
        mov al,0Ch
        out 00h,al
        mov al,04h
        out 02h,al
        call DELAY_20ms
        mov al,00h
        out 02h,al

        mov al,06h
        out 00h,al
        call DELAY_20ms
        mov al,04h
        out 02h,al
        call DELAY_20ms
        mov al,00h
        out 02h,al
        mov al,4ch
        out 00h,al
        call    DELAY_20ms          ;LCD
INITIALIZATION ENDS
```

```
   mov ax,0200h
        mov ds,ax

        mov si,0000h
        mov al,0bdh          ;hard  coding
pass-word ; 9763106626976310
        mov [si],al

        mov al,0d7h
        mov [si+1],al

        mov al,0dbh
        mov [si+2],al

        mov al,0e7h
        mov [si+3],al

        mov al,0edh
        mov [si+4],al

        mov al,0eeh
        mov [si+5],al

        mov al,0dbh
        mov [si+6],al

        mov al,0dbh
        mov [si+7],al

        mov al,0ebh
        mov [si+8],al

        mov al,0dbh
        mov [si+9],al

        mov al,0bdh
        mov [si+0ah],al

        mov al,0d7h
        mov [si+0bh],al

        mov al,0dbh
        mov [si+0ch],al

        mov al,0e7h
        mov [si+0dh],al

        mov al,0edh
        mov [si+0eh],al

        mov al,0eeh
        mov [si+0fh],al

        add si,000fh
        inc si


        mov al,0bdh          ;hard  coding
alarm pass-word ; 99999999999999
        mov [si],al

        mov al,0bdh
        mov [si+1],al
```

```
            mov al,0bdh
            mov [si+2],al

            mov al,0bdh
            mov [si+3],al

            mov al,0bdh
            mov [si+4],al

            mov al,0bdh
            mov [si+5],al

            mov al,0bdh
            mov [si+6],al

            mov al,0bdh
            mov [si+7],al

            mov al,0bdh
            mov [si+8],al

            mov al,0bdh
            mov [si+9],al

            mov al,0bdh
            mov [si+0ah],al

            mov al,0bdh
            mov [si+0bh],al

            mov al,0bdh
            mov [si+0ch],al

            mov al,0bdh
            mov [si+0dh],al

            add si,000dh
   inc si

            mov al,0ffh
            out 08h,al
;add code here to display "Command Key"
start:      call clear_LCD
                        call welcome_msg



                        call keypad_input
                        cmp al,0bbh
                        jz master_mode
                        jmp start
 ;press valid key
;add code here to display 'Invalid key'

x6: call clear_LCD
            call welcome_msg
            call keypad_input
            cmp al,0b7h
            jz User_mode
            jmp x6 ;press valid key

master_mode:
  call intm
  cmp ax,0abcdh
  jnz x6
x8:   call keypad_input
```

```
        cmp al,7Dh
        jz Alarm_mode
        jnz x8

Alarm_mode:
  call inta
  cmp dh,6h
  jz start
  cmp dh,1h
  jz x6
  jmp x70

User_mode:
  call intu
  cmp ax,0abcdh
  jz x8
  jnz x6
        ;Intentionally Left Blank

x70:
stop: jmp stop

DELAY_20ms proc

        MOV             CH,5
        X4:     NOP
                NOP
                DEC     CH
                JNZ     X4
        RET
DELAY_20ms endp
DELAY_0.04s proc

        MOV             cx,4fffh
        X17:    NOP
                NOP
                DEC     cx
                JNZ     X17
        RET
DELAY_0.04s endp
DELAY_max proc

        MOV             cx,0ffffh
        X16:    NOP
                NOP
                DEC     cx
                JNZ     X16
        RET
DELAY_max endp
;DELAY_20ms
enter_LCD proc
        mov al,0A0h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints Space

        mov al,0A0h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints Space
```

```
mov al,50h
out 00h,al
call DELAY_20ms
mov al,05h
out 02h,al
call DELAY_20ms
mov al,01h
out 02h,al  ;prints P

mov al,52h
out 00h,al
call DELAY_20ms
mov al,05h
out 02h,al
call DELAY_20ms
mov al,01h
out 02h,al  ;prints R

mov al,45h
out 00h,al
call DELAY_20ms
mov al,05h
out 02h,al
call DELAY_20ms
mov al,01h
out 02h,al  ;prints E

mov al,53h
out 00h,al
call DELAY_20ms
mov al,05h
out 02h,al
call DELAY_20ms
mov al,01h
out 02h,al  ;prints S

mov al,53h
out 00h,al
call DELAY_20ms
mov al,05h
out 02h,al
call DELAY_20ms
mov al,01h
out 02h,al  ;prints S

mov al,0A0h
out 00h,al
call DELAY_20ms
mov al,05h
out 02h,al
call DELAY_20ms
mov al,01h
out 02h,al  ;prints Space

mov al,45h
out 00h,al
call DELAY_20ms
mov al,05h
out 02h,al
call DELAY_20ms
mov al,01h
out 02h,al  ;prints E

mov al,4Eh
out 00h,al
call DELAY_20ms
mov al,05h
out 02h,al
```

```
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints N

        mov al,54h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints T

        mov al,45h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints E

        mov al,52h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints R
RET
enter_LCD endp


intm proc

                call clear_LCD
                mov al,0feh
                out 08h,al   ;glow  enter
pass LED

        ;byte by byte pass enter
                mov cx,16

                ;store      the      16-bit
entered pass after the hard coded pass word

enter_16bit:
                call keypad_input
                cmp al,7eh
                jz pressc
                cmp al,7bh
                jz pressac
                cmp al,77h
                jz press_enter
                cmp al,0bbh
                jz nop_master
                cmp al,0b7h
                jz nop_master
                cmp al,7dh
                jz nop_master
                mov [si],al
                CALL Print_*
                inc si
                dec cx
                jnz enter_16bit
disp_entermaster:
                call keypad_input
                cmp al,7eh
```

```
                jz pressc
                cmp al,7bh
                jz pressac
                cmp al,77h
                jz press_enter
asd:    CALL clear_LCD
                CALL    enter_LCD
        ;add code here to display 'PRESS
ENTER' on lcd
                call keypad_input
                cmp al,77h
                jz press_enter
                jnz asd
nop_master: nop
                jmp enter_16bit

pressc:
                call clear_1digit_LCD
                dec si
                inc cx
                jmp enter_16bit
pressac:
                CALL clear_LCD
                mov cx,16
                mov si,1eh;start of pass
segment
                jmp enter_16bit
press_enter:
                CALL clear_LCD
                mov al,0ffh
                out 08h,al
                cmp cx,0
                jz cmp_pass
                jmp raise_alarm
                ;glow retry/update led
                ;byte by byte


day_pass:
                mov si,002Eh
                mov al,0fdh
                out 08h,al
                call DELAY_max
                call DELAY_max
                call DELAY_max
                call clear_LCD
                mov cx,12
enter_12bit:
                call keypad_input
                cmp al,7eh
                jz presscday
                cmp al,0bbh
                jz nop_day
                cmp al,0b7h
                jz nop_day
                cmp al,7dh
                jz nop_day
                cmp al,7bh
                jz pressacday
                cmp al,77h
                jz press_enterday
                mov [si],al
                CALL Print_*
                inc si
                dec cx
                jnz enter_12bit

disp_enter:
                                call
keypad_input
```

```
                                cmp al,7eh
                                jz presscday
                                cmp al,7bh
                                jz
pressacday
                                cmp al,77h
                                jz
press_enterday
asd1:           CALL clear_LCD
                                CALL
enter_LCD
        ;add code here to display 'PRESS
ENTER' on lcd
                                call
keypad_input
                                cmp al,77h
                                jz
press_enterday
                                jnz asd1
nop_day:nop
                jmp enter_12bit


presscday:
                call clear_1digit_LCD
                dec si
                inc cx
                jmp enter_12bit
pressacday:
                CALL clear_LCD
                jmp day_pass
press_enterday:
                CALL clear_LCD
                mov al,0ffh
                out 08h,al
                cmp cx,0
                jnz err_msg
                mov al,0fbh
                out 08h,al


                call DELAY_max
                call DELAY_max



                mov al,0ffh
                out 08h,al
                jz end_69h
err_msg:
                call error_msg
                jmp day_pass
cmp_pass:
                cld
                mov si,0000h
                mov di,001Eh
                mov cx,17
x5:             mov al,[si]
                mov bl,[di]
                dec cx
                jz day_pass
                cmp al,bl
                jnz raise_alarm
                inc si
                inc di
                jmp x5



raise_alarm:
                mov dh,5h
                mov al,0fh
                out 08h,al
```

```
                mov ax,0abcdh
end_69h:
ret
intm endp

Print_* proc
                mov al,2Ah
                out 00h,al
                call DELAY_20ms
                mov al,05h
                out 02h,al
                call DELAY_20ms
                mov al,01h
                out 02h,al  ;prints *
ret
Print_* endp

clear_LCD proc
        mov al,00h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        ;Clear Display
        out 00h,al
        call DELAY_20ms
        mov al,04h
        out 02h,al
        call DELAY_20ms
        mov al,00h
        out 02h,al
RET
clear_LCD endp

keypad_input proc ;SubR for keypad entry,al
has unique key input value.
x0:             mov al,00h
                out 04h,al
x1:             in al,04h
                and al,0f0h
                cmp al,0f0h
                jnz x1
                CALL DELAY_20ms

                mov al,00h
                        ;  Check  for
key press
                out 04,al
x2:             ;in al,0ch
                ;cmp al,0fbh
                ;jz sta

                in al,04h
                and al,0F0h
                cmp al,0F0h
                jz x2
                CALL DELAY_20ms

                mov al,00h
                        ;  Check  for
key press
                out 04,al
                in al,04h
                and al,0F0h
                cmp al,0F0h
                jz x2

                mov al,0Eh
                ;Check  for  key  press
column 1
```

```
                        mov bl,al
                        out 04h,al
                        in al,04h
                        and al,0f0h
                        cmp al,0f0h
                        jnz x3

                        mov al,0Dh
                                ;Check     for
key press column 2
                        mov bl,al
                        out 04h,al
                        in al,04h
                        and al,0f0h
                        cmp al,0f0h
                        jnz x3

                        mov al,0Bh
                                ;Check     for
key press column 3
                        mov bl,al
                        out 04h,al
                        in al,04h
                        and al,0f0h
                        cmp al,0f0h
                        jnz x3

                        mov al,07h
                                ;Check     for
key press column 4
                        mov bl,al
                        out 04h,al
                        in al,04h
                        and al,0f0h
                        cmp al,0f0h
                        jz x2

x3:                     or al,bl
ret
keypad_input endp

inta proc
        mov al,00eh
        out 08h,al



        mov cx,14
        mov si,3ah ;store       the       16-bit
entered pass after the hard coded pass word

enter_14bit:
                        call keypad_input
                        cmp al,7eh
                        jz pressc_alarm
                        cmp al,0bbh
                        jz nop_alarm
                        cmp al,0b7h
                        jz nop_alarm
                        cmp al,7dh
                        jz nop_alarm
                        cmp al,7bh
                        jz pressac_alarm
                        cmp al,77h
                        jz press_enter_alarm
                        mov [si],al
                        CALL Print_*
                        inc si
```

```
                        dec cx
                        jnz enter_14bit

disp_enteralarm:
                        call keypad_input
                        cmp al,7eh
                        jz pressc_alarm
                        cmp al,7bh
                        jz pressac_alarm
                        cmp al,77h
                        jz press_enter_alarm
asd2:       CALL clear_LCD
                        CALL            enter_LCD
            ;add code here to display 'PRESS
ENTER' on lcd
                        call keypad_input
                        cmp al,77h
                        jz press_enter_alarm
                        jnz asd2
nop_alarm: nop
                jmp enter_14bit
pressc_alarm:
                        call clear_1digit_LCD
                        dec si
                        inc cx
                        jmp enter_14bit
pressac_alarm:
                        call clear_LCD
                        mov cx,14
                        mov si,3ah;start of pass
segment
                        jmp enter_14bit
press_enter_alarm:
                        CALL clear_LCD
                        mov al,0fh
                        out 08h,al
                        cmp cx,0
                        jz cmp_pass_alarm
                        jnz x56
cmp_pass_alarm:
                        cld
                        mov si,10h
                        mov di,3ah
                        mov cx,14
                        repe cmpsb
                        cmp cx,00h
                        jnz x56
                        mov al,0ffh
                        out 08h,al
                        add dh,1h
x56:
ret
inta endp



intu proc
                        call clear_LCD
                        mov     dl,1    ;flag   for
checking two inputs
                        mov al,0feh
                out 08h,al
                        mov cx,12
                mov si,48h              ;store      the
16-bit entered pass after the hard coded
pass word
enter_12bitu:
                        call keypad_input
                        cmp al,7eh
                        jz pressc_user
```

```
                cmp al,7bh
                jz pressac_user
                cmp al,0bbh
                jz nop_user
                cmp al,0b7h
                jz nop_user
                cmp al,7dh
                jz nop_user
                cmp al,77h
                jz press_enter_user
                mov [si],al
                CALL Print_*
                inc si
                dec cx
                jnz enter_12bitu
disp_enter_user:
                call keypad_input
                cmp al,7eh
                jz pressc_user
                cmp al,7bh
                jz pressac_user
                cmp al,77h
                jz press_enter_user
asd3:   CALL clear_LCD
                CALL enter_LCD
        ;add code here to display 'PRESS
ENTER' on lcd
                call keypad_input
                cmp al,77h
                jz press_enter_user
                jnz asd3
nop_user:
                        nop
                        jmp
enter_12bitu
pressc_user:
                call clear_1digit_LCD
                dec si
                inc cx
                jmp enter_12bitu
pressac_user:
                call clear_LCD
                mov cx,12
                mov si,48h;start of pass
segment
                jmp enter_12bitu
press_enter_user:
                mov al,0ffh
                out 08h,al
                cmp cx,0
                jz cmp_pass_user
                jnz wrong_pass

cmp_pass_user:
                cld
                mov si,2eh
                mov di,48h
                mov cx,12
                repe cmpsb
                cmp cx,00h
                jnz wrong_pass
                jz open_door_user

wrong_pass :
                call clear_LCD
                mov si,48h
                mov cx,12
                cmp dl,0
                jz raise_alarm_user
                mov al,0fdh
                out 08h,al
```

```
                call retry_msg
                call DELAY_max
                call DELAY_max
                call clear_LCD
                mov cx,12
                dec dl
                jmp enter_12bitu
raise_alarm_user:
                mov dh,0
                mov al,0fh
                out 08h,al
                mov ax,0abcdh
                jmp end_70h
open_door_user:
                call open_door
end_70h:
ret
intu endp

ints proc

                call open_door

                ; CALL DELAY_0.04s
                ; mov al,00h
                ; out 0ch,al
ret
ints endp

open_door proc
        call clear_LCD
        mov al,8ah
        out 0Ah,al

        call DELAY_20ms

        mov al,0ah
        out 0Ah,al

x31:    in al,0ch
                cmp al,0ffh
                jnz x31
                call DELAY_20ms
                call close_door

ret
open_door endp

close_door proc
        mov al,03h
        out 0Ah,al
        call DELAY_max
        call DELAY_max
        call DELAY_max
        call DELAY_max
        call DELAY_max
        call DELAY_max
        call DELAY_max
        call DELAY_max
        call DELAY_max
        call DELAY_max
        call DELAY_max
        call DELAY_max
        call DELAY_max
        call DELAY_max
        call DELAY_max
        call DELAY_max
        call DELAY_max
        call DELAY_max
```

```
		call DELAY_max
		call DELAY_max
		call DELAY_max
		call DELAY_max
		call DELAY_max
		call DELAY_max
		call DELAY_max
		call DELAY_max


ret
close_door endp


welcome_msg proc
		mov al,0A0h
		out 00h,al
		call DELAY_20ms
		mov al,05h
		out 02h,al
		call DELAY_20ms
		mov al,01h
		out 02h,al  ;prints Space

		mov al,0A0h
		out 00h,al
		call DELAY_20ms
		mov al,05h
		out 02h,al
		call DELAY_20ms
		mov al,01h
		out 02h,al  ;prints Space

		mov al,0A0h
		out 00h,al
		call DELAY_20ms
		mov al,05h
		out 02h,al
		call DELAY_20ms
		mov al,01h
		out 02h,al  ;prints Space

		mov al,0A0h
		out 00h,al
		call DELAY_20ms
		mov al,05h
		out 02h,al
		call DELAY_20ms
		mov al,01h
		out 02h,al  ;prints Space

		mov al,0A0h
		out 00h,al
		call DELAY_20ms
		mov al,05h
		out 02h,al
		call DELAY_20ms
		mov al,01h
		out 02h,al  ;prints Space

		mov al,57h
		out 00h,al
		call DELAY_20ms
		mov al,05h
		out 02h,al
		call DELAY_20ms
		mov al,01h
		out 02h,al  ;prints W
```

```
        mov al,45h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints E

        mov al,4Ch
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints L

        mov al,43h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints C

        mov al,4Fh
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints O

        mov al,4dh
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints M

        mov al,45h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints E

ret
welcome_msg endp

update_msg proc
        mov al,55h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints U
```

```
mov al,50h
out 00h,al
call DELAY_20ms
mov al,05h
out 02h,al
call DELAY_20ms
mov al,01h
out 02h,al  ;prints P

mov al,44h
out 00h,al
call DELAY_20ms
mov al,05h
out 02h,al
call DELAY_20ms
mov al,01h
out 02h,al  ;prints D

mov al,41h
out 00h,al
call DELAY_20ms
mov al,05h
out 02h,al
call DELAY_20ms
mov al,01h
out 02h,al  ;prints A

mov al,54h
out 00h,al
call DELAY_20ms
mov al,05h
out 02h,al
call DELAY_20ms
mov al,01h
out 02h,al  ;prints T

mov al,45h
out 00h,al
call DELAY_20ms
mov al,05h
out 02h,al
call DELAY_20ms
mov al,01h
out 02h,al  ;prints E

mov al,0A0h
out 00h,al
call DELAY_20ms
mov al,05h
out 02h,al
call DELAY_20ms
mov al,01h
out 02h,al  ;prints Space

mov al,50h
out 00h,al
call DELAY_20ms
mov al,05h
out 02h,al
call DELAY_20ms
mov al,01h
out 02h,al  ;prints P

mov al,41h
out 00h,al
call DELAY_20ms
mov al,05h
out 02h,al
call DELAY_20ms
```

```
        mov al,01h
        out 02h,al  ;prints A

        mov al,53h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints S

        mov al,53h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints S

        mov al,57h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints W

        mov al,4Fh
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints O

        mov al,52h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints R

        mov al,44h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints D

ret
update_msg endp

clear_1digit_LCD proc
        mov al,00h
        out 02h,al
        call DELAY_20ms
        mov al,10h
        ;shift left by 1
        out 00h,al
        call DELAY_20ms
        mov al,04h
```

```
        out 02h,al
        call DELAY_20ms
        mov al,00h
        out 02h,al

        mov al,0A0h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al
        ;prints Space

        call DELAY_20ms
        mov al,10h
        ;shift left by 1
        out 00h,al
        call DELAY_20ms
        mov al,04h
        out 02h,al
        call DELAY_20ms
        mov al,00h
        out 02h,al

RET
clear_1digit_LCD endp

error_msg proc
        mov al,0A0h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints Space

        mov al,45h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints E

        mov al,4Eh
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints N

        mov al,54h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints T

        mov al,45h
        out 00h,al
        call DELAY_20ms
```

```
mov al,05h
out 02h,al
call DELAY_20ms
mov al,01h
out 02h,al  ;prints E

mov al,52h
out 00h,al
call DELAY_20ms
mov al,05h
out 02h,al
call DELAY_20ms
mov al,01h
out 02h,al  ;prints R

mov al,0A0h
out 00h,al
call DELAY_20ms
mov al,05h
out 02h,al
call DELAY_20ms
mov al,01h
out 02h,al  ;prints Space

mov al,31h
out 00h,al
call DELAY_20ms
mov al,05h
out 02h,al
call DELAY_20ms
mov al,01h
out 02h,al  ;prints 1

mov al,32h
out 00h,al
call DELAY_20ms
mov al,05h
out 02h,al
call DELAY_20ms
mov al,01h
out 02h,al  ;prints 2

mov al,0A0h
out 00h,al
call DELAY_20ms
mov al,05h
out 02h,al
call DELAY_20ms
mov al,01h
out 02h,al  ;prints Space

mov al,44h
out 00h,al
call DELAY_20ms
mov al,05h
out 02h,al
call DELAY_20ms
mov al,01h
out 02h,al  ;prints D

mov al,49h
out 00h,al
call DELAY_20ms
mov al,05h
out 02h,al
call DELAY_20ms
mov al,01h
out 02h,al  ;prints I
```

```
        mov al,47h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints G

        mov al,49h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints I

        mov al,54h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints T

        mov al,53h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints S
RET
error_msg endp

retry_msg proc
        mov al,0A0h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints Space

        mov al,0A0h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints Space

        mov al,0A0h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints Space

        mov al,0A0h
        out 00h,al
        call DELAY_20ms
```

```
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints Space

        mov al,52h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints R

        mov al,45h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints E

        mov al,54h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints T

        mov al,52h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints R

        mov al,59h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints Y

ret
retry_msg endp


updateday_msg proc
        mov al,55h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints U

        mov al,50h
        out 00h,al
        call DELAY_20ms
```

```
mov al,05h
out 02h,al
call DELAY_20ms
mov al,01h
out 02h,al  ;prints P

mov al,44h
out 00h,al
call DELAY_20ms
mov al,05h
out 02h,al
call DELAY_20ms
mov al,01h
out 02h,al  ;prints D

mov al,41h
out 00h,al
call DELAY_20ms
mov al,05h
out 02h,al
call DELAY_20ms
mov al,01h
out 02h,al  ;prints A

mov al,54h
out 00h,al
call DELAY_20ms
mov al,05h
out 02h,al
call DELAY_20ms
mov al,01h
out 02h,al  ;prints T

mov al,45h
out 00h,al
call DELAY_20ms
mov al,05h
out 02h,al
call DELAY_20ms
mov al,01h
out 02h,al  ;prints E

mov al,0a0h
out 00h,al
call DELAY_20ms
mov al,05h
out 02h,al
call DELAY_20ms
mov al,01h
out 02h,al  ;prints Space

mov al,44h
out 00h,al
call DELAY_20ms
mov al,05h
out 02h,al
call DELAY_20ms
mov al,01h
out 02h,al  ;prints D

mov al,41h
out 00h,al
call DELAY_20ms
mov al,05h
out 02h,al
call DELAY_20ms
mov al,01h
out 02h,al  ;prints A
```

```
        mov al,59h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints Y

        mov al,0a0h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints Space

        mov al,50h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al ;prints P

        mov al,41h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints A

        mov al,53h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints S

        mov al,53h
        out 00h,al
        call DELAY_20ms
        mov al,05h
        out 02h,al
        call DELAY_20ms
        mov al,01h
        out 02h,al  ;prints S
ret
updateday_msg endp

Nmi_24hrtimer:

        call clear_LCD

        call clear_1digit_LCD

        call updateday_msg
startnmi:

                call keypad_input
                cmp al,0bbh
                jz master_mode
                jmp startnmi
```

```
iret
Switch_intR:
        call open_door
        ;iret
        sti
        jmp x6
```