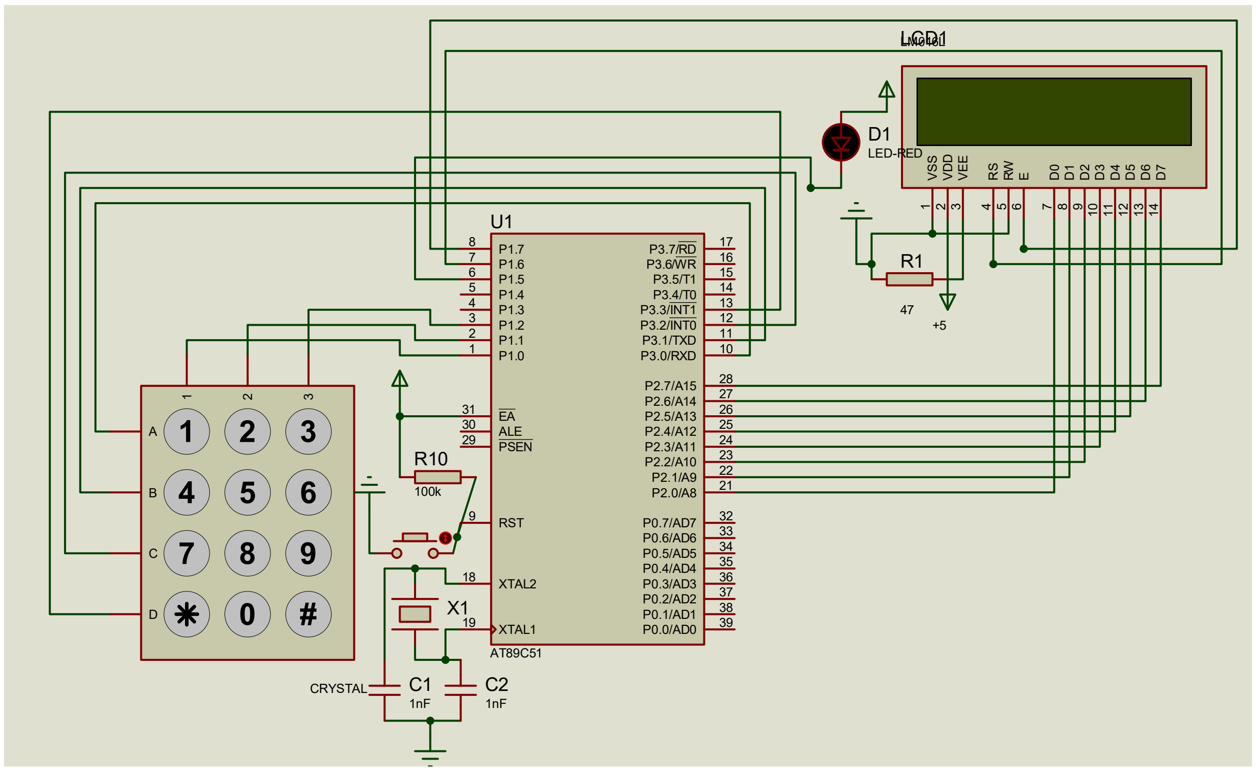
Project Name: To make a Security code lock by microcontroller AT89c51 and using keyboard and LCD .

Solution:

In this project , our work in only softer based. Here first we make a default password 1111.than we use a change option .We can change this password. And here enter a open this lock option .where we are enter the password and open this lock.

Hardware Design:



Software Algorithm:

1.Here we first initialization LCD

2.Than we show our program menu .

3. Here program menu,1.change password 2.Enter the lock.

4.Here default password is 1111.

5. Here we can change the password ,when we enter the change password option.

6.whwn we enter the 1. change password, first it said enter the old password,
Than if we give correct password, they give new password option .here we are give new password

7.when we go to 2. Enter the lock Option, here give the correct password, then the lock open,
If we give wrong password this lock can not open. If we give wrong password in 3 time this softer is looked. Than we make a reset the program.

Software Code:

```
DTA EQU P2
EN EQU P3.7
```

```
RS EQU P3.6
led equ p3.5
key_row equ p3
key_col equ p1

 ACALL INIT_LCD
 ACALL LCD_CLR
 ACALL WORK_6
 ACALL START
D_3: ACALL WORK_1  ;1.CHANGE PASSWORD
 ACALL WORK_2   ;2.ENTER THE LOCK
 ACALL KEYREE
 ACALL LOCK_WORK2
 ACALL LCD_CLR
 SJMP D_3
 D_4: ACALL WORK_4   ;OLD PASSEWORD
 MOV R0,#050H
D_6: ACALL KEYREE
 ACALL LOCK_WORK3
 SJMP D_6
 D_7: ACALL WORK_5   ;NEW PASSEWORD
D_8: ACALL KEYREE
 ACALL LOCK_WORK4
 SJMP D_8

D_11:
 MOV R2,#03H
D_1: ACALL WORK_3    ; ENTER PASSEWORD
 MOV R0,#050H
D_2: ACALL keyree
 ACALL LOCK_WORK1
 SJMP D_2

WORK_1: MOV DTA, #080h
  ACALL SEND_CMD
  MOV DTA, #'1'
  ACALL SEND_TXT
  MOV DTA, #'.'
  ACALL SEND_TXT
  MOV DTA, #'C'
  ACALL SEND_TXT
  MOV DTA, #'H'
  ACALL SEND_TXT
  MOV DTA, #'A'
  ACALL SEND_TXT
  MOV DTA, #'N'
  ACALL SEND_TXT
  MOV DTA, #'G'
  ACALL SEND_TXT
  MOV DTA, #'E'
  ACALL SEND_TXT
  MOV DTA, #' '
  ACALL SEND_TXT
  MOV DTA, #'P'
  ACALL SEND_TXT
  MOV DTA, #'A'
  ACALL SEND_TXT
  MOV DTA, #'S'
  ACALL SEND_TXT
  MOV DTA, #'S'
  ACALL SEND_TXT
  MOV DTA, #'W'
  ACALL SEND_TXT
  MOV DTA, #'O'
  ACALL SEND_TXT
  MOV DTA, #'R'
  RET

WORK_2:
  MOV DTA, #0C0h
  ACALL SEND_CMD
  MOV DTA, #'2'
  ACALL SEND_TXT
  MOV DTA, #'.'
  ACALL SEND_TXT
  MOV DTA, #'E'
  ACALL SEND_TXT
  MOV DTA, #'N'
  ACALL SEND_TXT
  MOV DTA, #'T'
  ACALL SEND_TXT
  MOV DTA, #'E'
  ACALL SEND_TXT
  MOV DTA, #'R'
  ACALL SEND_TXT
  MOV DTA, #' '
  ACALL SEND_TXT
  MOV DTA, #'T'
  ACALL SEND_TXT
  MOV DTA, #'H'
  ACALL SEND_TXT
  MOV DTA, #'E'
  ACALL SEND_TXT
  MOV DTA, #' '
  ACALL SEND_TXT
  MOV DTA, #'L'
  ACALL SEND_TXT
  MOV DTA, #'O'
  ACALL SEND_TXT
  MOV DTA, #'C'
  ACALL SEND_TXT
  MOV DTA, #'K'
```

```
 ACALL SEND_TXT
 RET


WORK_3:
 ACALL LCD_CLR
 MOV DTA, #080h
 ACALL SEND_CMD
 MOV DTA, #'E'
 ACALL SEND_TXT
 MOV DTA, #'N'
 ACALL SEND_TXT
 MOV DTA, #'T'
 ACALL SEND_TXT
 MOV DTA, #'E'
 ACALL SEND_TXT
 MOV DTA, #'R'
 ACALL SEND_TXT
 MOV DTA, #' '
 ACALL SEND_TXT
 MOV DTA, #'P'
 ACALL SEND_TXT
 MOV DTA, #'A'
 ACALL SEND_TXT
 MOV DTA, #'S'
 ACALL SEND_TXT
 MOV DTA, #'S'
 ACALL SEND_TXT
 MOV DTA, #'E'
 ACALL SEND_TXT
 MOV DTA, #'W'
 ACALL SEND_TXT
 MOV DTA, #'O'
 ACALL SEND_TXT
 MOV DTA, #'R'
 ACALL SEND_TXT
 MOV DTA, #'D'
 ACALL SEND_TXT
 MOV DTA, #':'
 ACALL SEND_TXT
 MOV R5,#0C4h
 RET

 WORK_4:
 MOV DTA, #080h
 ACALL SEND_CMD
 MOV DTA, #'O'
 ACALL SEND_TXT
 MOV DTA, #'L'
 ACALL SEND_TXT
 MOV DTA, #'D'
```

```
ACALL SEND_TXT
MOV DTA, #' '
ACALL SEND_TXT
MOV DTA, #'P'
ACALL SEND_TXT
MOV DTA, #'A'
ACALL SEND_TXT
MOV DTA, #'S'
ACALL SEND_TXT
MOV DTA, #'S'
ACALL SEND_TXT
MOV DTA, #'E'
ACALL SEND_TXT
MOV DTA, #'W'
ACALL SEND_TXT
MOV DTA, #'O'
ACALL SEND_TXT
MOV DTA, #'R'
ACALL SEND_TXT
MOV DTA, #'D'
ACALL SEND_TXT
MOV DTA, #':'
ACALL SEND_TXT
MOV R5,#0C4h
RET

 WORK_5:
MOV DTA, #080h
ACALL SEND_CMD
MOV DTA, #'N'
ACALL SEND_TXT
MOV DTA, #'E'
ACALL SEND_TXT
MOV DTA, #'W'
ACALL SEND_TXT
MOV DTA, #' '
ACALL SEND_TXT
MOV DTA, #'P'
ACALL SEND_TXT
MOV DTA, #'A'
ACALL SEND_TXT
MOV DTA, #'S'
ACALL SEND_TXT
MOV DTA, #'S'
ACALL SEND_TXT
MOV DTA, #'E'
ACALL SEND_TXT
MOV DTA, #'W'
ACALL SEND_TXT
MOV DTA, #'O'
ACALL SEND_TXT
```

```
  MOV DTA, #'R'
  ACALL SEND_TXT
  MOV DTA, #'D'
  ACALL SEND_TXT
  MOV DTA, #':'
  ACALL SEND_TXT
  MOV R5,#0C4h
  RET


WORK_6: MOV DTA, #084h
  ACALL SEND_CMD
  MOV DTA, #'W'
  ACALL SEND_TXT
  MOV DTA, #'E'
  ACALL SEND_TXT
  MOV DTA, #'L'
  ACALL SEND_TXT
  MOV DTA, #'L'
  ACALL SEND_TXT
  MOV DTA, #'C'
  ACALL SEND_TXT
  MOV DTA, #'O'
  ACALL SEND_TXT
  MOV DTA, #'M'
  ACALL SEND_TXT
  MOV DTA, #'E'
  ACALL SEND_TXT
   MOV DTA, #0C6h
  ACALL SEND_CMD
  MOV DTA, #'A'
  ACALL SEND_TXT
  MOV DTA, #'J'
  ACALL SEND_TXT
  MOV DTA, #'A'
  ACALL SEND_TXT
  MOV DTA, #'Y'
  ACALL SEND_TXT
  MOV R7,#02FH
LL1: ACALL DELAY
  DJNZ R7,LL1
  RET

;WORK KEYBORD........
keyree: MOV  key_col,#0FFH             ;make key_col an input port
K1: MOV  key_row,#00H                  ;ground all rows at once
            MOV  A,key_col             ;read all col(ensure keys open)
            ANL  A,#00000111B          ;masked unused bits
            CJNE A,#00000111B,K1       ;till all keys release
K2: ACALL DELAY                       ;call 20 msec delay
            MOV  A,key_col             ;see if any key is pressed
```

```
          ANL  A,#00000111B              ;mask unused bits
          CJNE A,#00000111B,OVER         ;key pressed, find row
          SJMP K2 ;check till key pressed
OVER: ACALL DELAY                        ;wait 20 msec debounce time
          MOV  A,key_col                 ;check key closure
          ANL  A,#00000111B              ;mask unused bits
          CJNE A,#00000111B,OVER1        ;key pressed, find row
          SJMP K2                        ;if none, keep polling
OVER1: MOV key_row, #11111110B   ;ground row 0
          MOV A,key_col                 ;read all columns
          ANL A,#00000111B              ;mask unused bits
          CJNE A,#00000111B,ROW_0  ;key row 0, find col.
          MOV key_row,#11111101B      ;ground row 1
          MOV A,key_col                 ;read all columns
          ANL A,#00000111B               ;mask unused bits
          CJNE A,#00000111B,ROW_1  ;key row 1, find col.
          MOV key_row,#11111011B       ;ground row 2
          MOV A,key_col                   ;read all columns
          ANL A,#00000111B                ;mask unused bits
          CJNE A,#00000111B,ROW_2   ;key row 2, find col.
          MOV key_row,#11110111B       ;ground row 3
          MOV A,key_col                  ;read all columns
          ANL A,#0000111B                ;mask unused bits
          CJNE A,#0000111B,ROW_3     ;key row 3, find col.
          LJMP K2 ;if none, false input,   ;repeat
ROW_0: MOV  DPTR,#KCODE0          ;set DPTR=start of row 0
          SJMP FIND                      find col. Key belongs to
ROW_1: MOV  DPTR,#KCODE1           ;set DPTR=start of row
          SJMP FIND                       ;find col. Key belongs to
ROW_2: MOV  DPTR,#KCODE2          ;set DPTR=start of row 2
          SJMP FIND                        ;find col. Key belongs to
ROW_3: MOV  DPTR,#KCODE3          ;set DPTR=start of row 3
FIND: RRC  A                             ;see if any CY bit low
          JNC  MATCH                      ;if zero, get ASCII code
          INC  DPTR                      ;point to next col. addr
          SJMP FIND                      ;keep searching
MATCH: CLR  A                            ;set A=0 (match is found)
          MOVC A,@A+DPTR
          RET
;..............................
;HERE WORK IN OPEN LOCK
LOCK_WORK1:
          CJNE A,#1,L22
           MOV DTA, A
          ACALL SEND_CMD
          LJMP D_3
L22:      CJNE A,#2,L23
          DEC R0
          MOV A,@R0
          CJNE A,043H,G_2
          DEC R0
```

```
        MOV A,@R0
        CJNE A,042H,G_2
        DEC R0
        MOV A,@R0
        CJNE A,041H,G_2
        DEC R0
        MOV A,@R0
        CJNE A,040H,G_2
        ACALL LCD_CLR
        MOV DTA, #087h
        ACALL SEND_CMD
        MOV DTA, #'O'
        ACALL SEND_TXT
        MOV DTA, #'K'
        ACALL SEND_TXT
        CLR led
        SJMP $
        G_2:
        ACALL WORK_11
        MOV R7,#01FH
LL3: ACALL DELAY
        DJNZ R7,LL3
        DJNZ R2,G_5
        ACALL LCD_CLR
        MOV DTA, #087h
        ACALL SEND_CMD
        MOV DTA, #'L'
        ACALL SEND_TXT
        MOV DTA, #'O'
        ACALL SEND_TXT
        MOV DTA, #'C'
        ACALL SEND_TXT
        MOV DTA, #'K'
        ACALL SEND_TXT
        SJMP $
G_5:    LJMP D_1
L23:
     MOV DTA, R5
    ACALL SEND_CMD
    MOV DTA, A
    ACALL SEND_TXT
     ACALL DELAY
     inc R5
     SUBB A,#30H
     MOV @R0,A
     INC R0
     RET

WORK_11:
        ACALL LCD_CLR
        MOV DTA, #080h
```

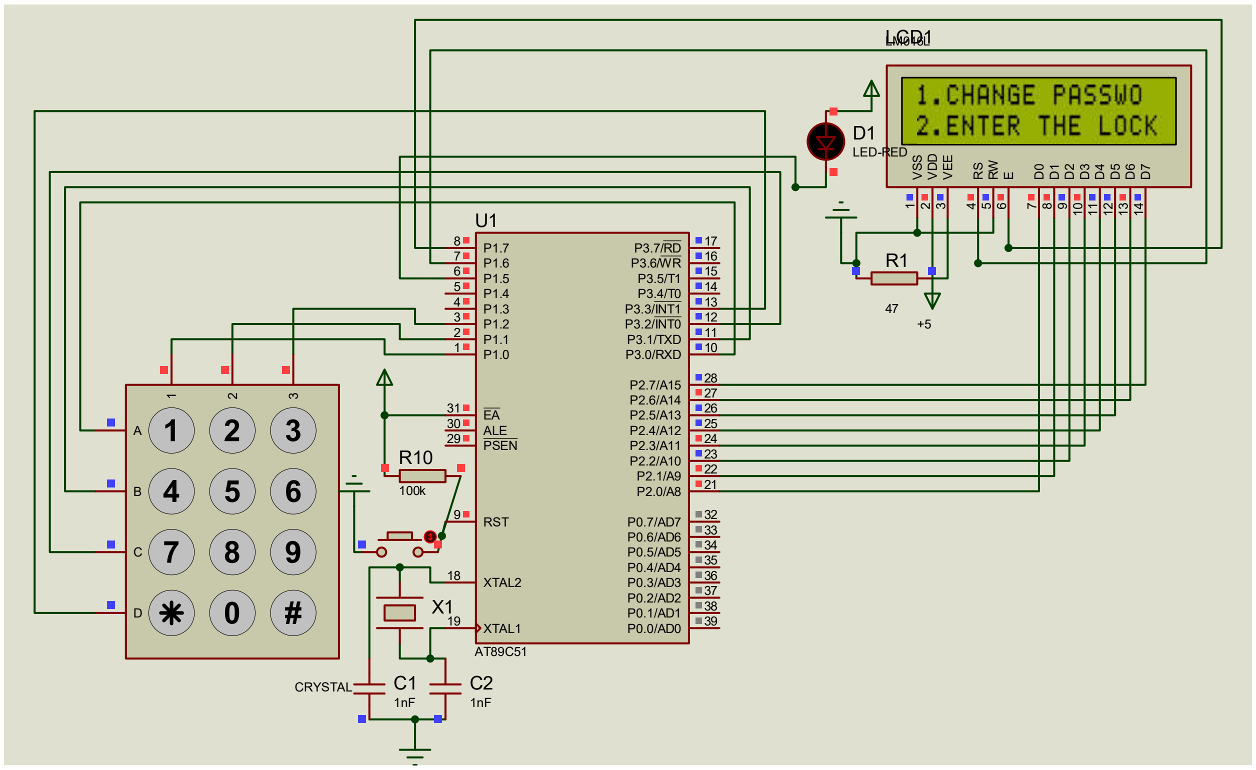
```
            ACALL SEND_CMD
            MOV DTA, #'W'
            ACALL SEND_TXT
            MOV DTA, #'R'
            ACALL SEND_TXT
            MOV DTA, #'O'
            ACALL SEND_TXT
            MOV DTA, #'N'
            ACALL SEND_TXT
            MOV DTA, #'G'
            ACALL SEND_TXT
            MOV DTA, #' '
            ACALL SEND_TXT
            MOV DTA, #'P'
            ACALL SEND_TXT
            MOV DTA, #'A'
            ACALL SEND_TXT
            MOV DTA, #'S'
            ACALL SEND_TXT
            MOV DTA, #'S'
            ACALL SEND_TXT
            MOV DTA, #'E'
            ACALL SEND_TXT
            MOV DTA, #'W'
            ACALL SEND_TXT
            MOV DTA, #'O'
            ACALL SEND_TXT
            MOV DTA, #'R'
            ACALL SEND_TXT
            MOV DTA, #'D'
            ACALL SEND_TXT
            MOV DTA, #'!'
            ACALL SEND_TXT
            RET             ;display pressed key
;......................
;PROGRAM CHOCE
LOCK_WORK2:
            SUBB A,#30H
            CJNE A,#1,L42
             ACALL LCD_CLR
        LJMP D_4
L42:        CJNE A,#2,L43
            ACALL LCD_CLR
            LJMP D_11
L43:        RET


;.....................
;OLD ENTER PASSOWED

LOCK_WORK3: CJNE A,#1,L52
```

```
             MOV DTA, A
            ACALL SEND_CMD
G_1:         LJMP D_3
L52:        CJNE A,#2H,L63
            DEC R0
            MOV A,@R0
            CJNE A,043H,G_1
            DEC R0
            MOV A,@R0
            CJNE A,042H,G_1
            DEC R0
            MOV A,@R0
            CJNE A,041H,G_1
            DEC R0
            MOV A,@R0
            CJNE A,040H,G_1
            ACALL LCD_CLR
            MOV R1,#040H
            LJMP D_7

L63:
     MOV DTA, R5
    ACALL SEND_CMD
    MOV DTA, A
    ACALL SEND_TXT
      ACALL DELAY
      inc R5
      SUBB A,#30H
      MOV @R0,A
      INC R0
      RET
;......................
;NEW ENTER PASSOWED

LOCK_WORK4: CJNE A,#1,L72
             MOV DTA, A
            ACALL SEND_CMD
        LJMP D_3
L72:        CJNE A,#2,L83
            ACALL LCD_CLR
            MOV DTA, #082h
            ACALL SEND_CMD
            MOV DTA, #'C'
            ACALL SEND_TXT
            MOV DTA, #'H'
            ACALL SEND_TXT
            MOV DTA, #'A'
            ACALL SEND_TXT
            MOV DTA, #'N'
            ACALL SEND_TXT
            MOV DTA, #'G'
            ACALL SEND_TXT
            MOV DTA, #'E'
            ACALL SEND_TXT
            MOV DTA, #' '
            ACALL SEND_TXT
            MOV DTA, #'O'
            ACALL SEND_TXT
            MOV DTA, #'K'
            ACALL SEND_TXT
            MOV R7,#01FH
LL2:    ACALL DELAY
            DJNZ R7,LL2
            LJMP D_3
L83:
     MOV DTA, R5
   ACALL SEND_CMD
   MOV DTA, A
   ACALL SEND_TXT
     ACALL DELAY
     inc R5
     SUBB A,#30H
     MOV @R1,A
     INC R1
     RET
;......................

LCD_CLR:   MOV DTA,#01h
   ACALL SEND_CMD
   RET
INIT_LCD:  MOV DTA,#38h
   ACALL SEND_CMD
   MOV DTA,#38h
   ACALL SEND_CMD
   MOV DTA,#38h
   ACALL SEND_CMD
   MOV DTA,#0Ch  ;Display on, cursor off
   ACALL SEND_CMD
   MOV DTA,#06h
   ACALL SEND_CMD
   MOV DTA,#80h  ;Line 1, Position 0
   ACALL SEND_CMD
   RET

SEND_CMD: CLR RS
   SETB EN
   CLR EN
   ACALL DELAY
   RET


DELAY:  MOV  R3,#255     ;50 or higher for fast CPUs
```

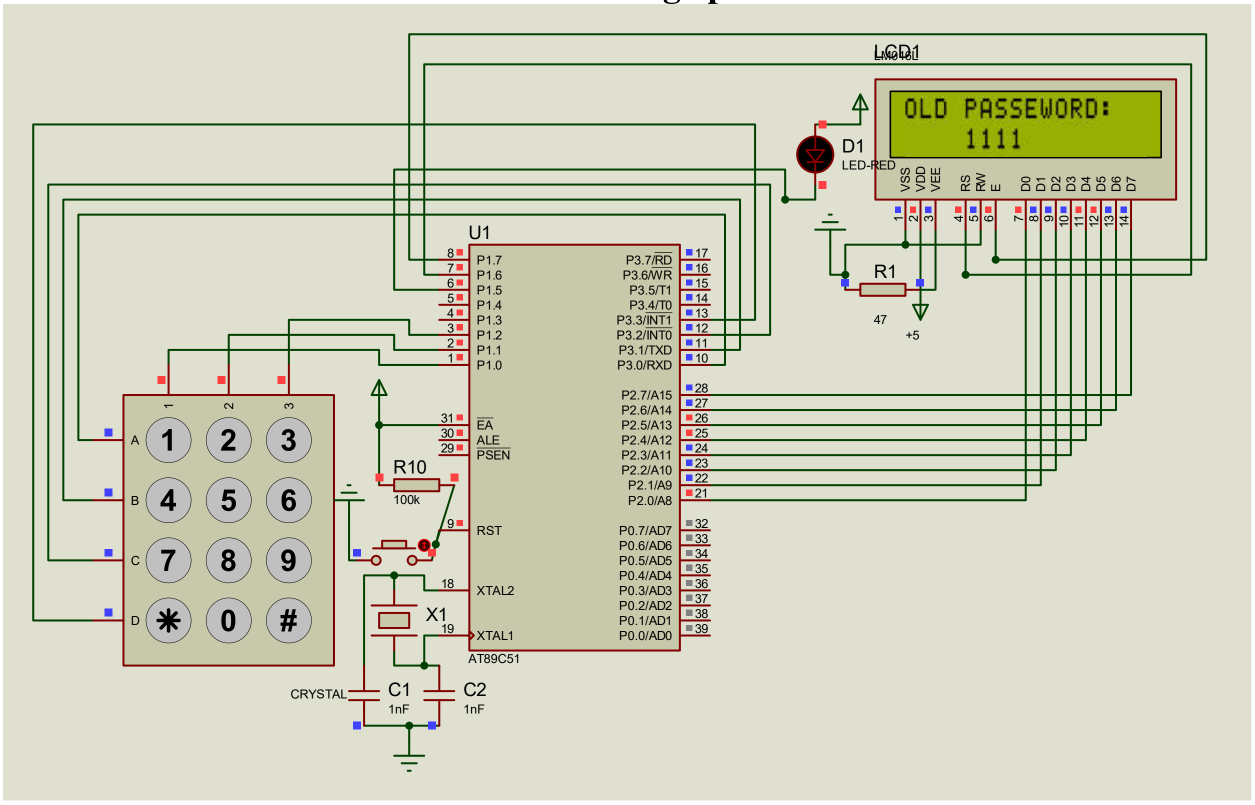
```
L2:     MOV  R4,#50    ;R4 = 255
L1:      DJNZ R4,L1    ;stay until R4 becomes
   DJNZ R3, L2
   RET
SEND_TXT:  SETB RS
   SETB EN
   CLR EN
   ACALL DELAY
   RET
START: MOV R1,#040H
 MOV R2,#04H
 D_5: MOV @R1,#01H
 INC R1
 DJNZ R2,D_5
 DEC R1
 RET
;ASCII LOOK-UP TABLE FOR EACH ROW
          ORG  0450H
KCODE0: DB   31H,32H,33H ;ROW 0
KCODE1: DB   34h,35h,36h ;ROW 1
KCODE2: DB   37h,38h,39h ;ROW 2
KCODE3: DB   1h,30h,2h ;ROW 3
          END
```

Output:

Main menu:

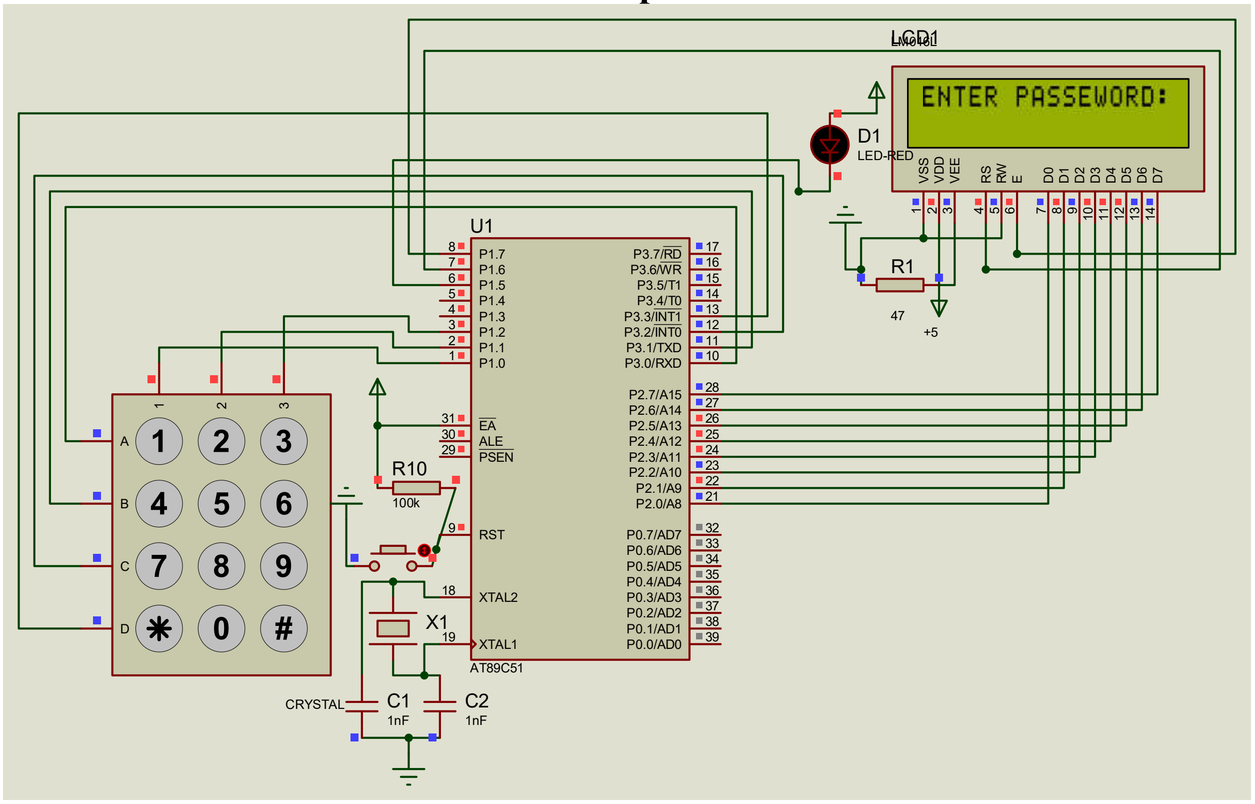


Change password:



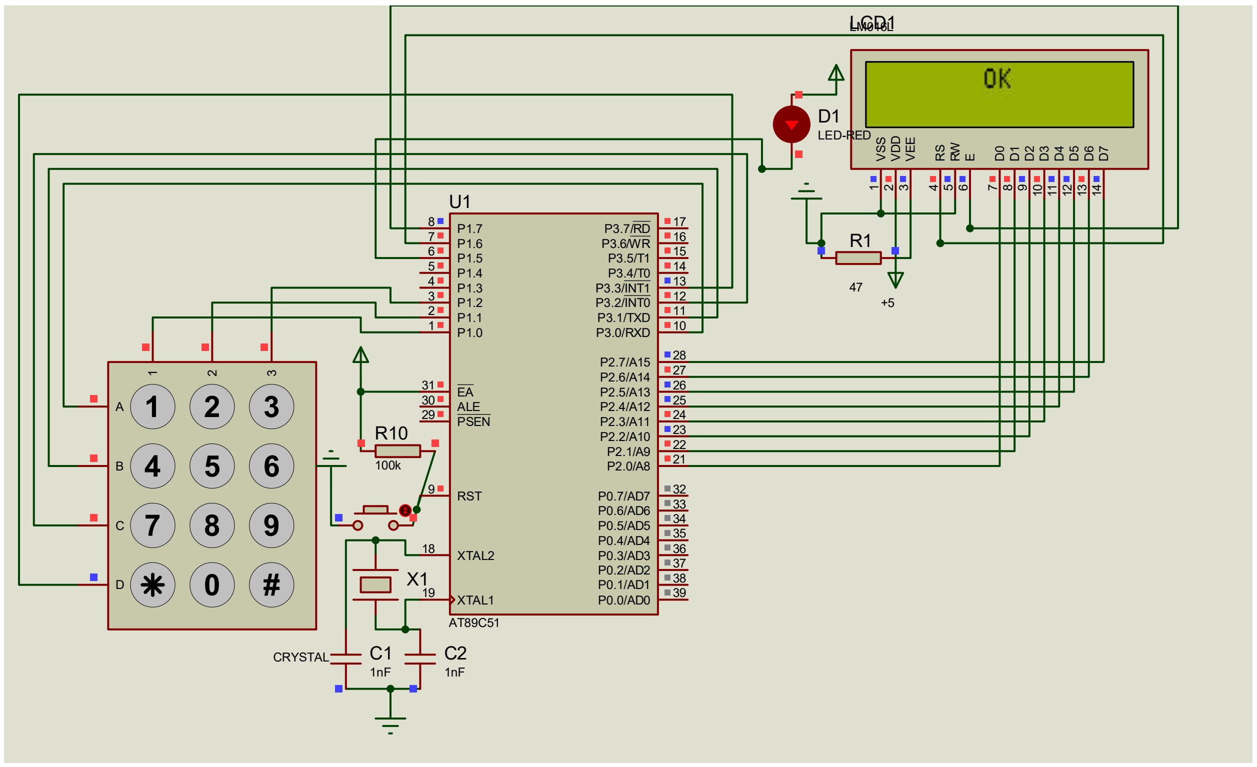
LCD1
LM016L
OLD PASSWORD:
1111
D1
LED-RED
VSS VDD VEE RS RW E D0 D1 D2 D3 D4 D5 D6 D7
R1
47
+5
U1
P1.7 P1.6 P1.5 P1.4 P1.3 P1.2 P1.1 P1.0
P3.7/RD P3.6/WR P3.5/T1 P3.4/T0 P3.3/INT1 P3.2/INT0 P3.1/TXD P3.0/RXD
P2.7/A15 P2.6/A14 P2.5/A13 P2.4/A12 P2.3/A11 P2.2/A10 P2.1/A9 P2.0/A8
P0.7/AD7 P0.6/AD6 P0.5/AD5 P0.4/AD4 P0.3/AD3 P0.2/AD2 P0.1/AD1 P0.0/AD0
EA ALE PSEN RST XTAL2 XTAL1
AT89C51
R10
100k
X1
CRYSTAL
C1
1nF
C2
1nF
A B C D
1 2 3 4 5 6 7 8 9 ✱ 0 #

Enter the password:



LCD1
LM016L
ENTER PASSWORD:
D1
LED-RED
VSS VDD VEE RS RW E D0 D1 D2 D3 D4 D5 D6 D7
R1
47
+5
U1
P1.7 P1.6 P1.5 P1.4 P1.3 P1.2 P1.1 P1.0
P3.7/RD P3.6/WR P3.5/T1 P3.4/T0 P3.3/INT1 P3.2/INT0 P3.1/TXD P3.0/RXD
P2.7/A15 P2.6/A14 P2.5/A13 P2.4/A12 P2.3/A11 P2.2/A10 P2.1/A9 P2.0/A8
P0.7/AD7 P0.6/AD6 P0.5/AD5 P0.4/AD4 P0.3/AD3 P0.2/AD2 P0.1/AD1 P0.0/AD0
EA ALE PSEN RST XTAL2 XTAL1
AT89C51
R10
100k
X1
CRYSTAL
C1
1nF
C2
1nF
A B C D
1 2 3 4 5 6 7 8 9 ✱ 0 #

Open lock:



Discussion: In this project we face some problem in ASM code. When we write scanning keyboard code we mach this code. We face another problem in designing . here when we use keyboard in port P0 ,it not work. So we connect LCD in P0 and P0 pull-up.

.