DSP1\_BUFFER EQU 030H ; For keep DSP1 Data

DSP2\_BUFFER EQU 031H ; For keep DSP2 Data

KPAD\_DATA EQU 032H ; For keep Keypad Data

Password\_field1 EQU 033H ; For password pin1

Password\_field2 EQU 034H ; For password pin2

Password\_field3 EQU 035H ; For password pin3

Password\_field4 EQU 036H ; For password pin4

FLAG EQU 02FH ; various purpose flag

DATA1 EQU 037H ; keypress store1

DATA2 EQU 038H ; keypress store2

DATA3 EQU 039H ; keypress store3

DATA4 EQU 03AH ; keypress store4

INPUT\_CHAR1 EQU 023H ;input character

INPUT\_CHAR3 EQU 024H ;input character

INPUT\_CHAR2 EQU 025H ;input character

;----------------------------------------------------------------------------

;Define Vectors for ISR

;----------------------------------------------------------------------------

ORG 0

LJMP MAIN ;3-byte instruction

LJMP EX0ISR ;03H EX0 vector

ORG 000BH ;Timer0 vector

LJMP T0ISR

ORG 001BH ;Timer1 vector

LJMP T1ISR

; ORG 0023H ;Serial Port vector

;LJMP SPISR

;----------------------------------------------------------------------------

; Main Program.

;----------------------------------------------------------------------------

ORG 003BH

MAIN: MOV P0,#0 ; Clear Databus

SETB P1.2 ; Clear DSP1

SETB P1.3 ; Clear DSP2

MOV P2,#11111111B ; Clear status keypad and 1-Wire

MOV R1,#DATA1

SETB IT0 ;logic low

MOV IE, #81H ; enable EXT 0 only

MOV FLAG,#0

;----------------------------------------------------------------------------

; Initial SCON and Bauad rate condition

;----------------------------------------------------------------------------

MOV SCON,#050H ; Mode1 RX Disable

MOV TMOD,#20H ; Set T1 to mode2

MOV TH1,#0FAH ;(-6)4800 bps Timer1

MOV TL1,#0FAH ;

SETB TR1

CLR TI ; Clear TI

CLR RI ; Clear RI

;----------------------------------------------------------------------------

; INITIALIZE MEM\_LOC And DSP

;----------------------------------------------------------------------------

MOV Data1,#0

MOV Data2,#0

MOV Data3,#0

MOV Data4,#0

SETB FLAG.4 ; password field blank?

MOV DSP1\_BUFFER,#1 ; Clear DSP1 Buffer

MOV DSP2\_BUFFER,#1 ; Clear DSP2 Buffer

;----------------------------------------------------------------------------

; Serial Port Interupt Service Routine

;----------------------------------------------------------------------------

;SPISR: JB TI,jsk

RISR: MOV INPUT\_CHAR3,INPUT\_CHAR2

MOV INPUT\_CHAR2,INPUT\_CHAR1

CLR RI

MOV A, SBUF

MOV INPUT\_CHAR1,A

MOV DSP2\_BUFFER,#12

MOV DSP1\_BUFFER,#12

ACALL SHOW\_DSP

ACALL Delay\_100ms

MOV A,#0

MOV DPTR,#CHAR1

MOVC A,@A+DPTR

CJNE A,INPUT\_CHAR3,LOOP

MOV A,#0

MOV DPTR,#CHAR2

MOVC A,@A+DPTR

CJNE A,INPUT\_CHAR2,LOOP

MOV A,#0

MOV DPTR,#CHAR3

MOVC A,@A+DPTR

CJNE A,INPUT\_CHAR1,LOOP

SETB FLAG.2

SJMP LOOP

;----------------------------------------------------------------------------

; Pin Enter

;----------------------------------------------------------------------------

LOOP: JB RI,RISR

lol: ACALL GET\_KPAD ; Get Keypad Data

MOV A,KPAD\_DATA

CJNE A,#0BH,Here

AJMP Warning

Here: CJNE A,#0DH,CheckBlank

AJMP SAVED

CheckBlank: JB FLAG.0,NEXTT ;to check if nothing enterd

Task1: JB FLAG.1 ,SHOW ; Check Keypad still pressed? KEYPRESSED

SETB FLAG.1 ; Set bit keypressed KEYPRESSED

CLR FLAG.4

DigitEntered:AJMP DIGITstore ;confirmed digit

LEFT\_SHIFT: MOV DSP2\_BUFFER,DSP1\_BUFFER ; Shift Display to Right

MOV DSP1\_BUFFER,KPAD\_DATA ; Shift Keypad data in

AJMP SHOW ; Jump to Show Display

NEXTT: CLR FLAG.1 ; Clear bit keypressed KEYPRESSED

SHOW: ACALL SHOW\_DSP ; Call Show DSP Subroutine

AJMP LOOP ; Jump to loop

;----------------------------------------------------------------------------

;Digit Storing

;----------------------------------------------------------------------------

DIGITstore: MOV DATA4,DATA3

MOV DATA3,DATA2

MOV DATA2,DATA1

MOV DATA1,KPAD\_DATA

SJMP LEFT\_SHIFT

;----------------------------------------------------------------------------

; Keypad Scan key Subroutine

;----------------------------------------------------------------------------

GET\_KPAD:CLR FLAG.0

MOV P2,#07FH ; Pull P2 to High

MOV KPAD\_DATA,#1 ; Clear Keypad Data

CHK\_COL0: CLR P2.6 ; Begin Scan Column 0 KPAD\_COL0

MOV A,P2 ; Get Port2 Value

ANL A,#00FH ; Get only lower 4 bit

CJNE A,#00FH,COL0\_DETECT ; Check All rows '1'?

AJMP CHK\_COL1 ; All rows '1' => check next column

COL0\_DETECT: MOV KPAD\_DATA,#02 ; Initial KPAD\_DATA = 1

AJMP GET\_ROW ; Jump to get row value

CHK\_COL1: SETB P2.6 ; Stop Scan Column 0 KPAD\_COL0

CLR P2.5 ; Begin Scan Column 1 KPAD\_COL1

MOV A,P2 ; Get Port2 Value

ANL A,#00FH ; Get only lower 4 bit

CJNE A,#00FH,COL1\_DETECT ; Check All rows '1'?

AJMP CHK\_COL2 ; All rows '1' => check next column

COL1\_DETECT: MOV KPAD\_DATA,#03 ; Initial KPAD\_DATA = 2

AJMP GET\_ROW ; Jump to get row value

CHK\_COL2: SETB P2.5 ; Stop Scan Column 1 KPAD\_COL1

CLR P2.4 ; Begin Scan Column 2 KPAD\_COL2

MOV A,P2 ; Get Port2 Value

ANL A,#00FH ; Get only lower 4 bit

CJNE A,#00FH,COL2\_DETECT ; Check All rows '1'?

SETB FLAG.0

RET ; All rows '1' => return

COL2\_DETECT: MOV KPAD\_DATA,#04 ; Initial KPAD\_DATA = 3

GET\_ROW:CLR P2.6 ; Enable all Column to find Crosspoint KPAD\_COL0

CLR P2.5 ; KPAD\_COL1

CLR P2.4 ;KPAD\_COL2

JB P2.0 ,CHK\_ROW1 ; Check Row 0 Detect?

RET ; Row 0 Detect => return

CHK\_ROW1: JB P2.1 ,CHK\_ROW2 ; Check Row 2 Detect? KPAD\_ROW1

MOV A,KPAD\_DATA ; Add 3 with KPAD\_DATA

ADD A,#3

MOV KPAD\_DATA,A

RET ; Return

CHK\_ROW2: JB P2.2, CHK\_ROW3 ; Check Row 2 Detect? KPAD\_ROW2

MOV A,KPAD\_DATA ; Add 6 with KPAD\_DATA

ADD A,#6

MOV KPAD\_DATA,A

RET ; Return

CHK\_ROW3: MOV A,KPAD\_DATA ; Add 9 with KPAD\_DATA

ADD A,#9

MOV KPAD\_DATA,A

RET ; Return

;----------------------------------------------------------------------------

; Show DSP Subroutine

;----------------------------------------------------------------------------

SHOW\_DSP: MOV R5,#5 ; Set loop 5 times

SCAN\_DSP\_LOOP: MOV A,DSP1\_BUFFER ; Restore DSP1 to Display

MOV DPTR,#DSP\_BLANK ; Move DIGIT Start Pointer

MOVC A,@A+DPTR ; Get ROM Data from Pointer+ACC.

MOV P0,A ; Out ACC. to DATABUS

CLR P1.2 ; Enable DSP1

ACALL DELAY\_10ms ; Delay

SETB P1.2 ; Disable DSP1

MOV A,DSP2\_BUFFER ; Restore DSP2 to Display

MOV DPTR,#DSP\_BLANK ; Move DIGIT Start Pointer

MOVC A,@A+DPTR ; Get ROM Data from Pointer+ACC.

MOV P0,A ; Out ACC. to DATABUS

CLR P1.3 ; Enable DSP2

ACALL DELAY\_10ms ; Delay

SETB P1.3 ; Disable DSP1

DJNZ R5,SCAN\_DSP\_LOOP; Do until 5 times

RET

;----------------------------------------------------------------------------

;Clear data field

;----------------------------------------------------------------------------

CLear\_Data\_field:

MOV Data1,#0

MOV Data2,#0

MOV Data3,#0 ;clear password field

MOV Data4,#0

SETB FLAG.4

CLR FLAG.5

CLR FLAG.6

CLR FLAG.7

RET

;----------------------------------------------------------------------------

;Show text on PC

;----------------------------------------------------------------------------

TX\_TEXT:CLR TR1

CLR TR0

MOV SCON,#050H ; Mode1 RX Disable

MOV TMOD,#20H ; Set T1 to mode2

MOV TH1,#0FAH ;(-6)4800 bps Timer1

MOV TL1,#0FAH ;

SETB TR1

CLR TI ; Clear TI

CLR RI ; Clear RI

;-----------------------------------------

TX\_LOOP: CLR A ; Clear ACC.

MOVC A,@A+DPTR ; Get Data from ROM with Pointer

INC DPTR ; Increase Pointer

CJNE A,#0,TX\_CHAR ; Check null terminated 000H Char.

RET ; End => Return

TX\_CHAR:MOV SBUF,A ; Send Data to SBUF

JNB TI,$ ; Wait until TX already (TI=1)

jsk: CLR TI ; Clear TI

ACALL DELAY\_100ms ; Delay each character

AJMP TX\_LOOP ; Jump to TX\_LOOP

;----------------------------------------------------------------------------

; Warnings (\*entered, blank password...)

;----------------------------------------------------------------------------

Warning:ACALL bell2

ACALL bell2

ACALL bell2

ACALL CLear\_Data\_field

MOV DPTR,#SERIAL\_TEXT ; Set Pointer Serial TX

ACALL TX\_TEXT ; TX Text to Serial Port

AJMP LOOP

;------------------------------------

Warning2:ACALL bell2

ACALL bell2

ACALL bell2

DPR0: JNB FLAG.4,DPR1

MOV DPTR,#SERIAL\_TEXT5 ; Set Pointer Serial TX

SJMP gg

DPR1: JNB FLAG.5,DPR2

MOV DPTR,#SERIAL\_TEXT6 ; Set Pointer Serial TX

SJMP gg

DPR2: JNB FLAG.6,DPR3

MOV DPTR,#SERIAL\_TEXT7 ; Set Pointer Serial TX

SJMP gg

DPR3: MOV DPTR,#SERIAL\_TEXT8 ; Set Pointer Serial TX

gg: ACALL CLear\_Data\_field

ACALL TX\_TEXT ; TX Text to Serial Port

AJMP LOOP

;-------------------------------------

Warning3:ACALL bell2

ACALL bell2

ACALL bell2

ACALL CLear\_Data\_field

MOV DPTR,#SERIAL\_TEXT10 ; Set Pointer Serial TX

ACALL TX\_TEXT ; TX Text to Serial Port

JumP: AJMP LOOP

;----------------------------------------------------------------------------

; Saved password

;----------------------------------------------------------------------------

SAVED:

JNB FLAG.2, Warning3

CheckPoint1:;check if all blank

JNB FLAG.4,CheckPoint2

AJMP Warning2

CheckPoint2:;check if only 1 pin entered

MOV A,DATA2

SETB FLAG.5

JZ Warning2

CheckPoint3:;check if only 2 pins entered

MOV A,DATA3

CLR FLAG.5

SETB FLAG.6

JZ Warning2

CheckPoint4:;check if only 3 pins entered

MOV A,DATA4

CLR FLAG.6

SETB FLAG.7

JZ Warning2

Store: MOV Password\_field1,Data4

MOV Password\_field2,Data3

MOV Password\_field3,Data2

MOV Password\_field4,Data1

ACALL bell1

ACALL bell1

ACALL bell1

ACALL CLear\_Data\_field

ACALL Temp

MOV DPTR,#SERIAL\_TEXT2 ; Set Pointer Serial TX

ACALL TX\_TEXT ; TX Text to Serial Port

JumP1: AJMP LOOP

;----------------------------------------------------------------------------

; Password comparison 1

;----------------------------------------------------------------------------

PC: MOV A,#0

MOV DPTR,#Password\_ID1

MOVC A,@A+DPTR

CJNE A,Data4,MisMatch

SJMP Checkk1

Checkk1: MOV A,#0

MOV DPTR,#Password\_ID2

MOVC A,@A+DPTR

CJNE A,Data3,MisMatch

SJMP Checkk2

Checkk2: MOV A,#0

MOV DPTR,#Password\_ID3

MOVC A,@A+DPTR

CJNE A,Data2,MisMatch

SJMP Checkk3

Checkk3: MOV A,#0

MOV DPTR,#Password\_ID4

MOVC A,@A+DPTR

CJNE A,Data1,MisMatch

SJMP Matched

;----------------------------------------------------------------------------

; Interrupt related program

;----------------------------------------------------------------------------

EX0ISR: JNB FLAG.2,PC

MOV A,Data4

CJNE A,Password\_field1,MisMatch

SJMP Check1

Check1: MOV A,Data3

CJNE A,Password\_field2,MisMatch

SJMP Check2

Check2: MOV A,Data2

CJNE A,Password\_field3,MisMatch

SJMP Check3

Check3: MOV A,Data1

CJNE A,Password\_field4,MisMatch

SJMP Matched

MisMatch:

JNB FLAG.4,jrk

MOV DPTR,#SERIAL\_TEXT9 ; Set Pointer Serial TX

SJMP caller

jrk: MOV DPTR,#SERIAL\_TEXT3 ; Set Pointer Serial TX

caller: ACALL TX\_TEXT ; TX Text to Serial Port

ACALL CLear\_Data\_field

CLR TR1

MOV TMOD, #11H ;16-bit timer mode

MOV R4, #40 ;40x50000 us = 2 second

SETB TF0 ;force Timer0 interrupt

SETB TF1 ;force Timer1 interrupt

SETB ET0 ;enable Timer0 interrupt

SETB ET1 ;enable Timer1 interrupt

RETI

T0ISR: CLR TR0 ;stop Timer0

DJNZ R4, SKIP ;if not 20th time, exit

CLR ET0 ;if 20th disable tone

CLR ET1 ;disable itself

MOV DPTR,#SERIAL\_TEXT12 ; Set Pointer Serial TX

ACALL TX\_TEXT ; TX Text to Serial Port

LJMP EXIT

SKIP: MOV TH0,#HIGH(-50000) ; 0.05 sec delay

MOV TL0,#LOW(-50000)

SETB TR0

EXIT: RETI

T1ISR: CLR TR1

MOV TH1,#HIGH(-1000) ;count for 400 Hz

MOV TL1,#LOW(-1000)

CPL P1.7 ; music

SETB TR1

RETI

;----------------------------------------------------------------------------

; If password Matched

;----------------------------------------------------------------------------

Matched:ACALL bell

ACALL bell

ACALL bell

ACALL CLear\_Data\_field

MOV DPTR,#SERIAL\_TEXT4 ; Set Pointer Serial TX

ACALL TX\_TEXT ; TX Text to Serial Port

AJMP LOOP

;----------------------------------------------------------------------------

; Bells, the blinks

;----------------------------------------------------------------------------

bell: MOV DSP2\_BUFFER,#0 ;Digit1 accepted

MOV DSP1\_BUFFER,#0

ACALL SHOW\_DSP

MOV DSP2\_BUFFER,#14 ;Digit1 accepted

MOV DSP1\_BUFFER,#14

ACALL SHOW\_DSP

RET

bell1: MOV DSP2\_BUFFER,#0 ;Digit1 accepted

MOV DSP1\_BUFFER,#0

ACALL SHOW\_DSP

MOV DSP2\_BUFFER,#11 ;Digit1 accepted

MOV DSP1\_BUFFER,#11

ACALL SHOW\_DSP

RET

bell2: MOV DSP2\_BUFFER,#0 ;Digit1 accepted

MOV DSP1\_BUFFER,#0

ACALL SHOW\_DSP

MOV DSP2\_BUFFER,#13 ;Digit1 accepted

MOV DSP1\_BUFFER,#13

ACALL SHOW\_DSP

RET

;----------------------------------------------------------------------------

; temp

;----------------------------------------------------------------------------

Temp: MOV DSP1\_BUFFER,Password\_field1

MOV DSP2\_BUFFER,Password\_field1

ACALL SHOW\_DSP

ACALL DELAY\_100ms

MOV DSP1\_BUFFER,Password\_field2

MOV DSP2\_BUFFER,Password\_field2

ACALL SHOW\_DSP

ACALL DELAY\_100ms

MOV DSP1\_BUFFER,Password\_field3

MOV DSP2\_BUFFER,Password\_field3

ACALL SHOW\_DSP

ACALL DELAY\_100ms

MOV DSP1\_BUFFER,Password\_field4

MOV DSP2\_BUFFER,Password\_field4

ACALL SHOW\_DSP

ACALL DELAY\_100ms

RET

;----------------------------------------------------------------------------

; Dummy Delay time 1m,10ms,1s

;----------------------------------------------------------------------------

DELAY\_10ms: MOV R7,#010 ; Do 10 times

DELAY\_10ms\_1: MOV R6,#0E6H ; Each loop = 1 ms

DELAY\_10ms\_2: NOP

NOP

DJNZ R6,DELAY\_10ms\_2

DJNZ R7,DELAY\_10ms\_1

RET

DELAY\_100ms: MOV R7,#100 ; Do 100 times

DELAY\_100ms\_1: MOV R6,#0E6H ; Each loop = 1 ms

DELAY\_100ms\_2: NOP

NOP

DJNZ R6,DELAY\_100ms\_2

DJNZ R7,DELAY\_100ms\_1

RET

;----------------------------------------------------------------------------

;Define Constant < Store in Flash EEPROM Program Memory >

;----------------------------------------------------------------------------

SERIAL\_TEXT: DB " Only digits Please Enter\_Again"

DB 0DH,0AH ,00H

SERIAL\_TEXT2: DB " Password Saved :)"

DB 0DH,0AH ,00H

SERIAL\_TEXT3: DB " Password MisMatch~OPEN :("

DB 0DH,0AH ,00H

SERIAL\_TEXT4: DB " Password Matched! :-) Reset the board"

DB 0DH,0AH ,00H

SERIAL\_TEXT5: DB " Password Field is blank, Enter 4 digits"

DB 0DH,0AH ,00H

SERIAL\_TEXT6: DB " You've entered only 1 pin enter 4 pins!!"

DB 0DH,0AH ,00H

SERIAL\_TEXT7: DB " You've entered only 2 pins enter 4 pins!!"

DB 0DH,0AH ,00H

SERIAL\_TEXT8: DB " You've entered only 3 pins enter 4 pins!!"

DB 0DH,0AH ,00H

SERIAL\_TEXT9: DB " Enter 4 pins before pressing S!!"

DB 0DH,0AH ,00H

SERIAL\_TEXT10: DB " Write me CAT before saving your own password "

DB 0DH,0AH ,00H

SERIAL\_TEXT12: DB " Try harder"

DB 0DH,0AH ,00H

; Segment .GFEDCBA

DSP\_BLANK: DB 00000000B

DSP\_DOT: DB 10000000B

DSP\_NUM1: DB 00000110B

DSP\_NUM2: DB 01011011B

DSP\_NUM3: DB 01001111B

DSP\_NUM4: DB 01100110B

DSP\_NUM5: DB 01101101B

DSP\_NUM6: DB 01111101B

DSP\_NUM7: DB 00000111B

DSP\_NUM8: DB 01111111B

DSP\_NUM9: DB 01101111B

DSP\_STAR: DB 01110110B

DSP\_NUM0: DB 00111111B

DSP\_HASH: DB 01100011B

DSP\_AA: DB 01110111B

PASSWORD\_ID1: DB 00000010B

PASSWORD\_ID2: DB 00000011B

PASSWORD\_ID3: DB 00000010B

PASSWORD\_ID4: DB 00000110B

CHAR1: DB "C"

CHAR2: DB "A"

CHAR3: DB "T"

END