Project Report

Group – 01 - B1

Temperature Controlled Motor

Contents

Topic ***Page***

Team Information

Member Name	Student ID
Md. Safin Mahmood Shomyo	190021217
Moshiur Rahman Chowdhury	190021225
Asif-ur-rahman Adib	190021227
Shafaiet Newaz Wornob	190021237
Mousouambi Djae Mbae	190021243

Objective

Project Overview: The primary objective of this project is to design a temperature-controlled motor using an 8051 microcontroller. The system incorporates a temperature sensor and keypad for user inputs. Keypad is used for taking maximum and minimum input that helps motor to operate as user wants. And two push buttons are used to select different mode of operations. Additionally, two extra features—temperature unit conversion and a motor timer—were envisioned.

Goals:

- Implement a robust temperature-controlled motor system.
- Taking user input of maximum and minimum temperature for the better control of the system.
- Ensuring users flexibility to select between different mode of operation (mode-1 and mode-2)
- Allow users to observe both Celsius and Fahrenheit as temperature display.
- Integrate a timer feature for the motor

Required Components

Components	Unit	Total Cost
MKS51 8051 Development Board	01	7500
DC Motor	01	50
Jumper Wires	50	50

Circuit Diagram

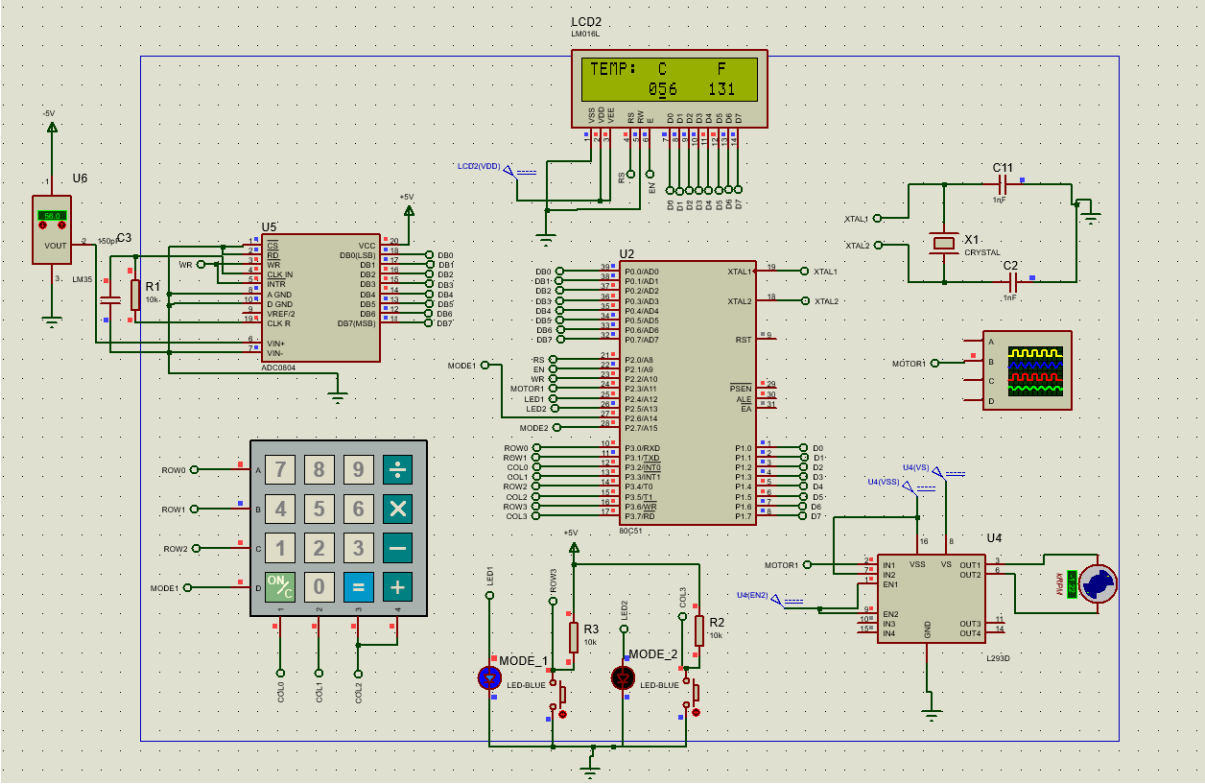


Fig: Circuit Diagram

Port 0

The whole port is connected with the ADC to take input

Port 1

The whole port is connected to the data port of LCD

Port 2

In p2.0 and p2.1 the RS and E pin of lcd is connected. In p2.2 the WR pin of the ADC is connected and p2.3 is giving pulse to the motor. P2.4 and p2.5 is for the led that shows the mode (Mode-1 or Mode-2). Pin2.6 is connected to the 4th row of keypad.

Port 3

In port 3 the p3.0 to p3.5 is connected to the keypad. And the p3.6 and p3.7 is for setting mode those are connected to push buttons.

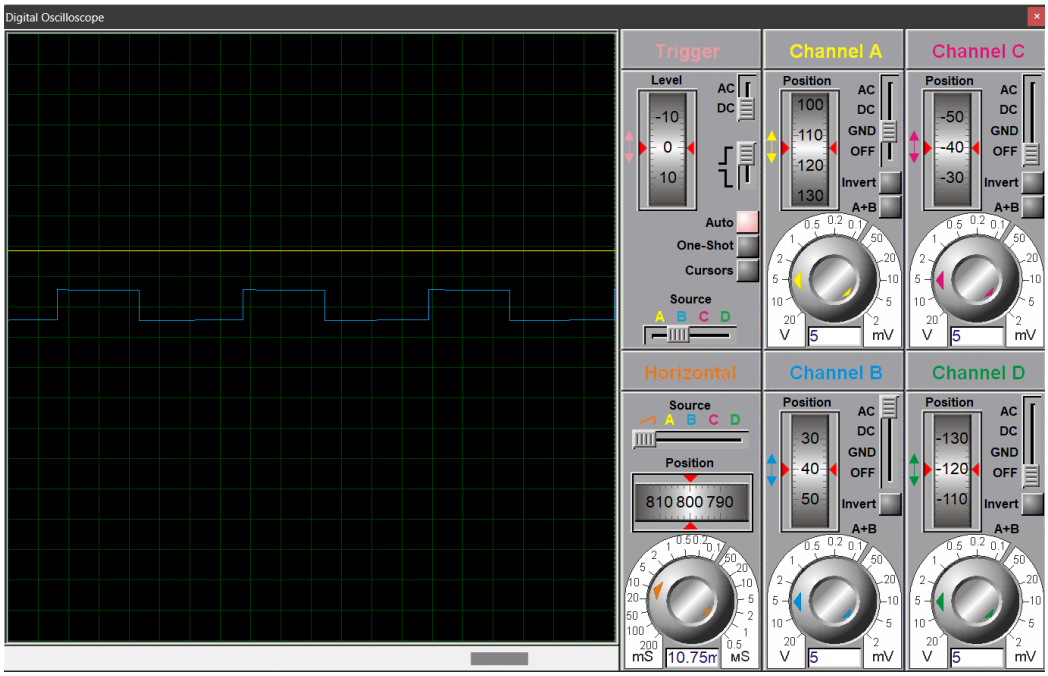


Fig: Output

If we focus on the hardware parameters, then we can see that, we would need to rotate the potentiometer to adjust the output pulses

Features

Mandatory Features:

1. Use temperature sensor to control any DC motor.

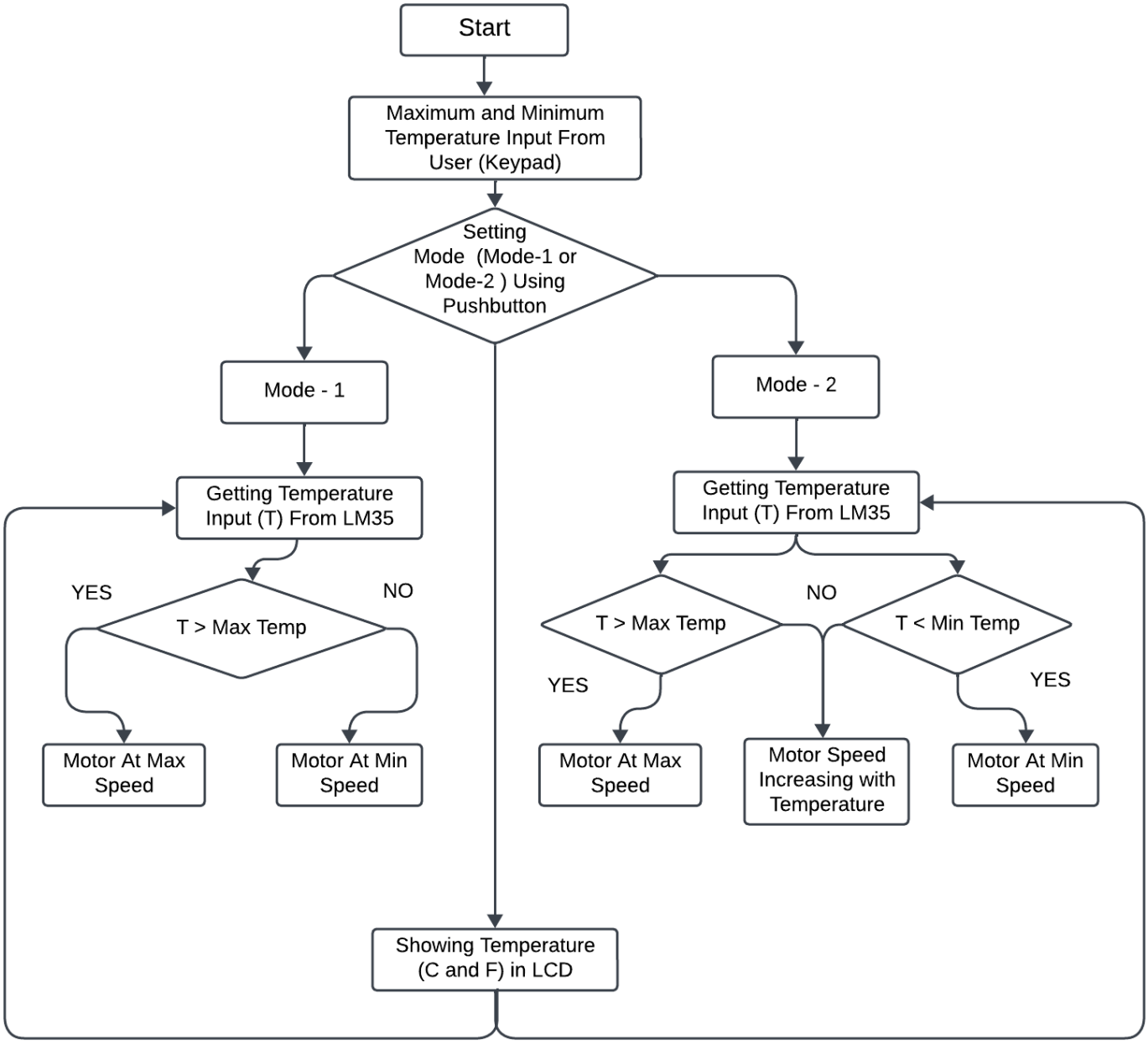
2. Take input using keypad. microcontroller will take the maximum and minimum temperature input from the user. Then microcontroller will take input. If "1" is pressed then motor will run at the maximum speed when temperature is greater than the given maximum temperature otherwise at minimum speed. If "2" is pressed motor will run according to the temperature in terms of given maximum and minimum temperature. For example: if min temperature =25 and max temperature =40. If temperature is lower than or equal to 25 RPM will run at min speed. If temperature is higher than max t temperature motor will run at max speed. If temperature is in between this range motor speed will gradually increase from the minimum speed or decrease from the maximum speed according to the temperature.

3. The temperature and speed must be shown on the LCD.

Additional Features:

2. **Temperature Unit Change:** Users can observe temperature in both Celsius and Fahrenheit unit.
3. **Motor Timer (Unimplemented):** Enable a timer for the motor that controls the duration of the motor being on.

Working Principle



Code

```
RS EQU P2.0
EN EQU P2.1
WRR EQU P2.2

DECI_RESULT EQU 90H
ASCII_RESULT EQU 95H

ORG     00H
LJMP    MAIN
ORG     000BH
LJMP    AFTER_INTR


MAIN:
MOV SP,#70H

MIN_MAX_DISPLAY:

     MOV A,#38H
     ACALL COMNWRT
     ACALL DELAY
     MOV A,#0EH
     ACALL COMNWRT
     ACALL DELAY
     MOV A,#01
     ACALL COMNWRT
     ACALL DELAY
     MOV A,#06
     ACALL COMNWRT
     ACALL DELAY
     MOV A,#80H
     ACALL COMNWRT
     ACALL DELAY


;::::::::::::MIN MAX INPUT TAKE::::::::::::::

 MOV DPTR, #SHOW1

 MIN_MAX_LOOP_1F:

        CLR A
        MOVC A,@A+DPTR
        JZ MIN_MAX_START1
        LCALL DATAWRT
```

```
        LCALL DELAY
        INC DPTR
        LJMP MIN_MAX_LOOP_1F


MIN_MAX_START1:
        CLR C
        CLR A
        SETB P3.2
        SETB P3.3
        SETB P3.5

        CLR P3.0
        CLR P3.1  ;
        CLR P3.4
        CLR p2.6
        MOV R2,#2



    MOV A,#0C0H   ; SHOWING IN SECOND LINE
     ACALL COMNWRT
     ACALL DELAY

     MOV A,#14H
     ACALL COMNWRT
     ACALL DELAY

     MOV A,#14H
     ACALL COMNWRT
     ACALL DELAY

     MOV A,#14H
     ACALL COMNWRT
     ACALL DELAY


K11:

                      CLR P3.0
                      CLR P3.1
                      CLR p3.4
                      CLR P2.6
                                        ;SENDING 0 TO THE KEYBOARD
                      JNB P3.2, K11
                      JNB P3.3, K11
                      JNB p3.5, K11           ;check till all keys released
K21:                  ACALL   DELAY                   ;call 20ms delay
```

```
                    JNB P3.2, OVERF1
                    JNB P3.3, OVERF1
                    JNB P3.5, OVERF1                    ;key pressed,
await closure

                    SJMP    K21                 ;check is key pressed
OVERF1:             ACALL   DELAY
                    JNB P3.2, OVER11             ;key pressed, find
row
                    JNB P3.3, OVER11
                    JNB P3.5, OVER11             ;key pressed, await
closure
                    SJMP    K21                  ;if none, keep
polling


OVER11:             CLR     P3.0
                    SETB    P3.1                ;ground row 0
                    SETB    P3.4
                    SETB    P2.6
                    JNB     P3.2,ROW_01         ;key row 0, find the
column
                    JNB     P3.3,ROW_01         ;key row 0, find the
column
                    JNB     P3.5,ROW_01

                    CLR     P3.1
                    SETB    P3.0
                    SETB    P3.4
                    SETB    P2.6
                    JNB     P3.2,ROW_11         ;key row 1, find the
column
                    JNB     P3.3,ROW_11         ;key row 1, find the
column
                    JNB     P3.5,ROW_11

                    CLR     P3.4
                    SETB    P3.1
                    SETB    P3.0
                    SETB    P2.6
                    JNB     P3.2,ROW_21         ;key row 1, find the
column
                    JNB     P3.3,ROW_21         ;key row 1, find the
column
                    JNB     P3.5,ROW_21

                    CLR     P2.6
                    SETB    P3.1
```

```
                        SETB    P3.0
                        SETB     P3.4
                        JNB     P3.2,ROW_31             ;key row 1, find the
column
                        JNB     P3.3,ROW_31             ;key row 1, find the
column
                        JNB     P3.5,ROW_31


                        LJMP    K21                     ;if none, false input,
repeat

ROW_01:                 MOV     DPTR, #KCODE0
                        SJMP    FIND11
ROW_11:                 MOV     DPTR, #KCODE1
                        SJMP    FIND11
ROW_21:                 MOV     DPTR, #KCODE2
                        SJMP    FIND11
ROW_31:                 MOV     DPTR, #KCODE3
                        SJMP    FIND11


FIND11:
                        JNB     P3.2,MATCH11
                        INC     DPTR
                        JNB     P3.3,MATCH11

                        INC     DPTR
                        JNB     P3.5,MATCH11

        MATCH11:


                        CLR     A                       ;set A=0 (match found)
                        MOVC    A, @A+DPTR              ;get ASCII code from
table
                        MOV R7,A
                        MOV B,A                       ;MOVC A,@A+DPTR
                        PUSH    7


                        LCALL   DATAWRT                 ;call display
subroutine


                        SKIPP:
                        MOV A,B
                        LCALL DELAY
```

```
                        DJNZ    R2,Kpapa
                        LCALL DELAY
                        LCALL DELAY

                        SJMP MIN_MAX_START1M
Kpapa: LJMP K11


;;;;;;;;;;;;;;MIN PART START HERE
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

MIN_MAX_START1M:   CLR C
        CLR A
        SETB P3.2 ;FOR INPUT
        SETB P3.3  ; FOR INPUT- FOR COLUMN
        SETB P3.5

        CLR P3.0
        CLR P3.1  ; FOR OUTPUT- FOR ROW
        CLR P3.4
        CLR P2.6
        MOV R2,#2



     MOV A,#14H
     ACALL COMNWRT
     ACALL DELAY

     MOV A,#14H
     ACALL COMNWRT
     ACALL DELAY

     MOV A,#14H
     ACALL COMNWRT
     ACALL DELAY

     MOV A,#14H
     ACALL COMNWRT
     ACALL DELAY


K11M:


                        CLR P3.0
                        CLR P3.1
```

```
                    CLR p3.4
                    CLR P2.6
                                        ;SENDING 0 TO THE KEYBOARD
                    JNB P3.2, K11M
                    JNB P3.3, K11M
                    JNB p3.5, K11M
K21M:               ACALL   DELAY
                    JNB P3.2, OVERF1M
                    JNB P3.3, OVERF1M
                    JNB P3.5, OVERF1M

                    SJMP    K21M
OVERF1M:                    ACALL   DELAY
                    JNB P3.2, OVER11M
                    JNB P3.3, OVER11M
                    JNB P3.5, OVER11M
                    SJMP    K21M


OVER11M:            CLR     P3.0
                    SETB    P3.1                ;ground row 0
                    SETB    P3.4
                    SETB    P2.6
                    JNB     P3.2,ROW_01M        ;key row 0, find the
column
                    JNB     P3.3,ROW_01M        ;key row 0, find the
column
                    JNB     P3.5,ROW_01M

                    CLR     P3.1
                    SETB    P3.0
                    SETB    P3.4
                    SETB    P2.6
                    JNB     P3.2,ROW_11M        ;key row 1, find the
column
                    JNB     P3.3,ROW_11M        ;key row 1, find the
column
                    JNB     P3.5,ROW_11M

                    CLR     P3.4
                    SETB    P3.1
                    SETB    P3.0
                    SETB    P2.6
                    JNB     P3.2,ROW_21M        ;key row 1, find the
column
                    JNB     P3.3,ROW_21M        ;key row 1, find the
column
                    JNB     P3.5,ROW_21M
```

```
                        CLR     P2.6
                        SETB    P3.4
                        SETB    P3.1
                        SETB    P3.0
                        JNB     P3.2,ROW_31M            ;key row 1, find the
column
                        JNB     P3.3,ROW_31M            ;key row 1, find the
column
                        JNB     P3.5,ROW_31M


                        LJMP    K21M

ROW_01M:                        MOV     DPTR, #KCODE0
                        SJMP    FIND11M
ROW_11M:                        MOV     DPTR, #KCODE1
                        SJMP    FIND11M
ROW_21M:                        MOV     DPTR, #KCODE2
                        SJMP    FIND11M
ROW_31M:                        MOV     DPTR, #KCODE3
                        SJMP    FIND11M


FIND11M:
                        JNB     P3.2,MATCH11M
                        INC     DPTR
                        JNB     P3.3,MATCH11M

                        INC     DPTR
                        JNB     P3.5,MATCH11M

        MATCH11M:
                        CLR     A
                        MOVC    A, @A+DPTR
                        MOV R7,A
                        MOV B,A
                        PUSH    7



                        LCALL   DATAWRT

                        SKIPPM:
                        MOV A,B
                        LCALL DELAY

                        DJNZ    R2,KpapaM
                        LCALL DELAY
```

```
                LCALL DELAY

                SJMP CHECK_MIN_MAX
KpapaM: LJMP K11M
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;



CHECK_MIN_MAX:
                POP     7
                POP     6
                POP     5
                pop     4


                MOV     A,R4
                SUBB    A,#30H
                Mov B,#10
                MUL AB
                MOV R4,A


                MOV     A,R5
                SUBB    A,#30H
                ADD A, R4

                SETB PSW.3
                MOV R1,A
                CLR PSW.3



                MOV     A,R6
                SUBB    A,#30H
                Mov B,#10
                MUL AB
                MOV R6,A


                MOV     A,R7
                SUBB    A,#30H
                ADD A, R6
                SETB PSW.3
                MOV R7,A
                CLR PSW.3
```

```
MAIN_PORTION:

MODE_SELECT:


; CLR P2.4
 ; CLR P2.5
  ;CLR P3.7
; MODE1:
 ;        JB P2.6, MODE2
  ;        CLR p2.4



;MODE2:    JB P2.7, MODE1
 ;         CLR P2.5



 MODE1:
         JB P3.6, MODE2
          CLR P2.4


                  MOV A,#01H
                ACALL COMNWRT
                ACALL DELAY
                MOV DPTR, #SHOW3
LOOP_02:        CLR A
                MOVC A,@A+DPTR
                JZ TEMPERATURE_WRITING ; WE HAVE TO CHANGE HERE
                LCALL DATAWRT
                LCALL DELAY
                INC DPTR
                LJMP LOOP_02


MODE2:    JB P3.7, MODE1
          CLR P2.5


                   MOV A,#01H
                ACALL COMNWRT
                ACALL DELAY
                MOV DPTR, #SHOW3
LOOP_03:        CLR A
                MOVC A,@A+DPTR
```

```
                JZ TEMPERATURE_WRITING ; WE HAVE TO CHANGE HERE
                LCALL DATAWRT
                LCALL DELAY
                INC DPTR
                LJMP LOOP_03


TEMPERATURE_WRITING:
     MOV A,#01
     ACALL COMNWRT
     ACALL DELAY



;;;;;;FIRST SENTENCE WRITING ( TEMP ;;;;;;;;;;;;;;;;

        MOV DPTR, #SHOW2
LOOP_1: CLR A
        MOVC A,@A+DPTR
        JZ INTR_TIMER ;
        LCALL DATAWRT
        LCALL DELAY
        INC DPTR
        LJMP LOOP_1

INTR_TIMER:
                MOV TMOD,#01H
                CLR P2.3 ;for pulse as output pin

                SETB P2.3
                CLR TF0
                MOV TH0,#0FFH;
                MOV TL0,#0FH
                MOV IE,#82H
                SETB TR0







;;;;;;;:::INITIALIZE THE ADC ::::::::::::::::::

TEMP:
          CLR WRR; WR

          SETB WRR; WR
```

```
;;;;;;;;For going to nex line ::::::::::::::::

     MOV A,#0C0H
     ACALL COMNWRT
     ACALL DELAY

     MOV A,#06H
     ACALL COMNWRT
     ACALL DELAY


;;;::::::::::::TEMPERATURE SHOW:::::::::::::::::::::::::::::;


TEMP_DIS:     MOV A,P0

           LCALL CONVERSION

           MOV A,#0C0H
           LCALL COMNWRT
           LCALL DELAY

           MOV A,#14H
           LCALL COMNWRT
           ;LCALL DELAY
           MOV A,#14H
           LCALL COMNWRT
           ;LCALL DELAY
           MOV A,#14H
           LCALL COMNWRT
           ;LCALL DELAY
           MOV A,#14H
           LCALL COMNWRT
           ;LCALL DELAY
           MOV A,#14H
           LCALL COMNWRT

           ;LCALL DELAY

                MOV A,#06H
     ACALL COMNWRT
     ACALL DELAY


           MOV R2,#4
```

```
     BACK: MOV A,@R1

           LCALL DATAWRT
           LCALL DELAY
           DEC R1
           DJNZ R2,BACK

              ;;;;:::::::::::::: For Showing the farenhite
temperature   ::::::::::::::;;;;;

              MOV A,#14H
           LCALL COMNWRT
           ;LCALL DELAY
           MOV A,#14H
           LCALL COMNWRT

          LCALL DELAY;

              MOV A,#06H
     ACALL COMNWRT
     ACALL DELAY

          LCALL CONVERSIONF

TEMP_DIS111:  MOV R2,#4
          ;MOV R1,#ASC_RESULT
     BACK111: MOV A,@R1

           LCALL DATAWRT
           LCALL DELAY
           DEC R1
           DJNZ R2,BACK111



                LCALL DELAY
                LCALL DELAY
                LCALL DELAY

               LJMP TEMP


AFTER_INTR:



 ;JNB P2.5, M1MODE
 ;JNB P2.4 ,M2MODE
```

```
 ;M1KAJER: CLR P3.7

  ;;;;;;;;;;;;; MODE-1 MODE-2 ;;;;;;;;;;;;;;;;;;;;;;;;;;

;M1MODE:
 ;
  ;                MOV A,P0
   ;               MOV B,#2
    ;              MUL AB
     ;             MOV R3,A

;CLR P3.7
;SETB PSW.3
;MOV B,R1
;CLR PSW.3
 ;         CJNE A, B, HE1
  ;HE1:    JNC  MAXX
   ;       SJMP MINN

;MAXX:     MOV A, #245  ;230
 ;         SJMP ORIGINAL

;MINN:
 ;      MOV A, #30
       ;CPL A
  ;     SJMP ORIGINAL



  JNB P2.4 , M1MODE
 JNB P2.5, M2MODE

 ;; MODE1 and MODe-2 er kaj niche


   M1MODE:           MOV A,P0
                MOV B,#2
                MUL AB
                MOV R5,A


SETB PSW.3
MOV B,#3

MOV A,R1
SUBB A,B
```

```
MOV B,A
CLR PSW.3
MOV A,R5
         CJNE A,B, HERE11V ; #45 here i will put the minimum temperature value
HERE11V:  JC LOWESTV
         SETB PSW.3
MOV B,R1
CLR PSW.3
         CJNE A,B, HERE22V ; #75 Here i will put the maximum temperature value
HERE22V:  JNC HIGHESTV
         SJMP INTERMEDIATE_SPEED



LOWESTV:   MOV A, #40
          SJMP ORIGINAL
HIGHESTV:  MOV A, #245  ;230
          SJMP ORIGINAL




M2MODE:           MOV A,P0
                  MOV B,#2
                  MUL AB
                  MOV R5,A


SETB PSW.3
MOV B,R7
CLR PSW.3

         CJNE A,B, HERE11 ; #45 here i will put the minimum temperature value
HERE11:  JC LOWEST
         SETB PSW.3
MOV B,R1
CLR PSW.3
         CJNE A,B, HERE22 ; #75 Here i will put the maximum temperature value
HERE22:  JNC HIGHEST
         SJMP INTERMEDIATE_SPEED



LOWEST:   MOV A, #40
          SJMP ORIGINAL
HIGHEST:  MOV A, #245  ;230
          SJMP ORIGINAL
```

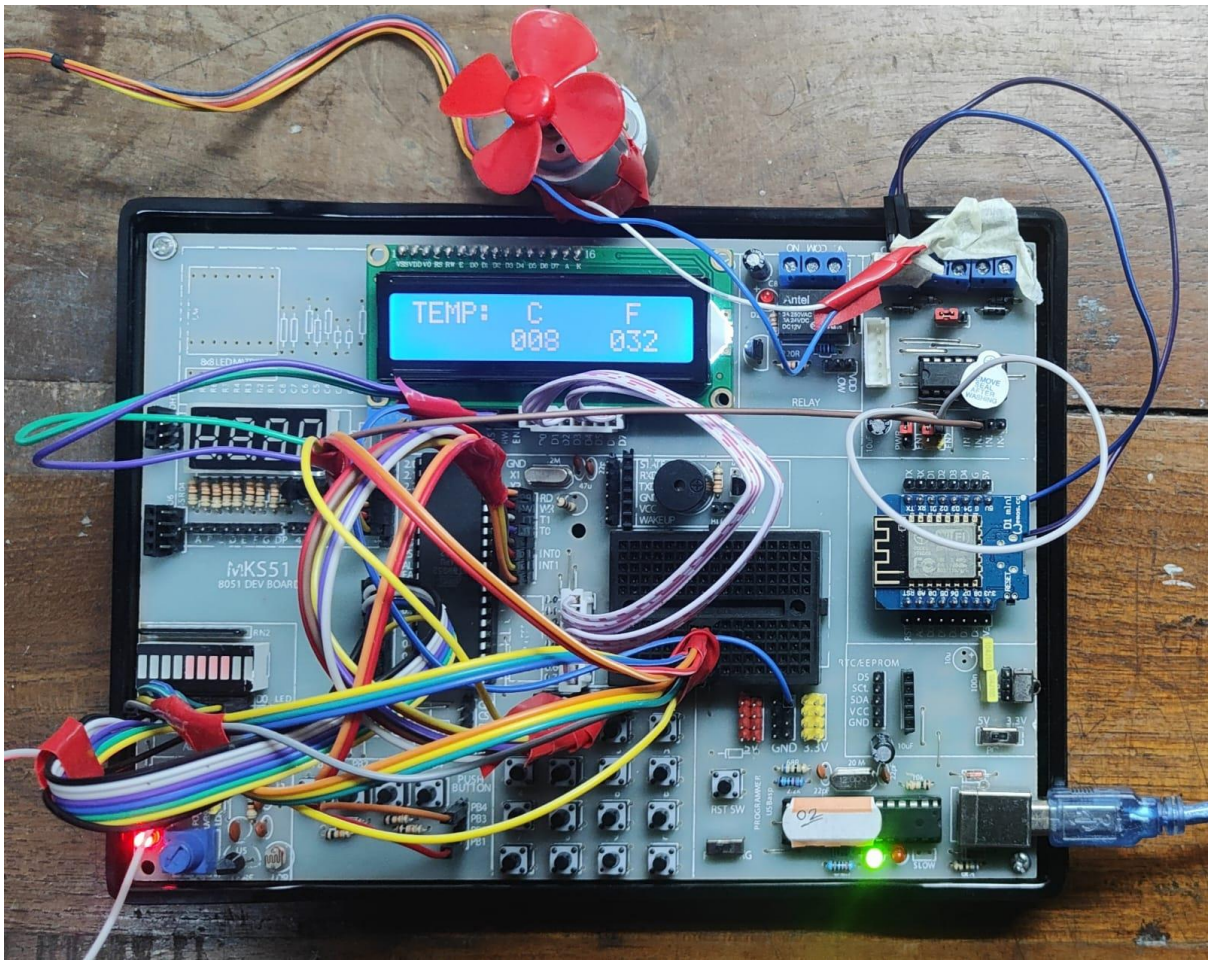
```
  INTERMEDIATE_SPEED:

  MOV A,R5    ; Ei line jokhon limit er moddhe ase tokhon A er value ferot
ane
  CLR C
  RLC A
  ;CPL A
  ORIGINAL :
  CPL A
                      MOV R7, A

                      LCALL LOW_DONE

                      SETB P2.3
                      CLR TF0
                      MOV TH0, R7; ; here CPL of A will be given. Lower TH0
represent that the value will be start from lower. that means more time it
will high.
                      SETB TR0
                      RETI


LOW_DONE:
                        CLR P2.3
                        MOV A, #0FFH ;careful here.
                        CLR C
                        SUBB A, R7
                        MOV TH0, A
STAY2:                  JNB TF0, STAY2
                        CLR TF0
                        RET



;;;;;;;::::DATA WRITE AND COMMAND WRITE::::::::::::

COMNWRT:
         MOV P1,A
         CLR RS
         SETB EN
         ACALL DELAY
         CLR EN
         RET

DATAWRT:
         MOV P1,A
         SETB RS
         SETB EN
```

```
          ACALL DELAY
          CLR EN
          RET

;;;;;;;;;;;;;:DELAY GENERATION FOR DISPLAY:::::::::::::
DELAY:
       MOV R3,#50
HERE2: MOV R4,#255
HERE1 : DJNZ R4,HERE1
       DJNZ R3,HERE2
       RET




;;;;;;;;;;;;;;;TEMPERATURE CONVERSION:::::::::::::::::::

CONVERSION:



BIN_DEC_CNVRT:
              MOV R0,#DECI_RESULT

              CLR C
              RLC A  ; Temperature double showing issue solved clr c and rlc a
line
              MOV B,#10
              DIV AB
              MOV @R0,B
              INC R0
              MOV B,#10
              DIV AB
              MOV @R0,B
              INC R0
              MOV @R0,A

DEC_ASC_CNVRT:

             MOV R0,#DECI_RESULT
             MOV R1,#ASCII_RESULT
             MOV R2,#3
      BACK1:  MOV A,@R0
             ORL A,#30H
             MOV @R1,A ; SAVE IT TO R1
             INC R0
             INC R1
             DJNZ R2,BACK1
             RET
```

```
CONVERSIONF:


BIN_DEC_CNVRTF:
               MOV R0,#DECI_RESULT
               MOV A,P0
               CLR C
               RLC A

                ;;;;;  CEL TO FER
              MOV B,#5
              DIV AB
              MoV B, #9
              MUL AB

              MOV B, #32
              ADD A,B
              ;;;; CLE TO FER


              ;;;;;
               MOV B,#10
               DIV AB
               MOV @R0,B
               INC R0
               MOV B,#10
               DIV AB
               MOV @R0,B
               INC R0
               MOV @R0,A

DEC_ASC_CNVRTF:

              MOV R0,#DECI_RESULT
              MOV R1,#ASCII_RESULT
              MOV R2,#3
      BACK1F:  MOV A,@R0
              ORL A,#30H
              MOV @R1,A ; SAVE IT TO R1
              INC R0
              INC R1
              DJNZ R2,BACK1F
              RET


;;;;;;;;DATA TO BE DISPLAYED:::::::::::::::
```

```
        ORG 500H


SHOW1: DB "    MAX | MIN:",0

SHOW2:DB"TEMP:  C      F",0
SHOW3:DB" ",0

KCODE0: DB      '7','8','9'                     ;ROW 0
KCODE1: DB      '4','5' ,'6'                    ;ROW 1
KCODE2: DB      '1', '2','3'
KCODE3: DB      '0', '0','0'

          END
```

Hardware Implementation



User inputs maximum and

Fig: Hardware Implementation

Based on user selection ("1" or "2"), the motor runs at either maximum or variable speed.

Motor speed adjusts based on the current temperature within the specified range.

Temperature and motor speed are displayed on the LCD.

Problems Faced

1. The memory location created problem. As the code of this project is very long will assigning proper memory location some problem raised. For example, same memory location is used for different purposes. In those cases, the result show in the lcd had one problem. LCD was showing arbitrary values.

2. Using timer for both motor speed control and our extra feaute motor timer (duration) created problem. As we needed interrupt for these features, we were able to apply interrupt for the mandatory feature motor speed control but couldn't apply it for the extra feature motor timer.

3. There are some issues related to the Machine cycle of the microcontroller 8051. In some cases, the software simulation it works properly but in hardware it takes some time or sometimes it stops working. For example, while taking input of maximum and minimum temperature we have to give input every time with a few seconds delays.

4. The Avrdudes software that is used for uploading the hex file in hardware created issue related to USB.

5. It is easier to observe small changes in software but it becomes very hard to observe those changes in hardware. For example, small changes in motor speed.

Conclusion

In conclusion, the 8051 microcontroller-based temperature-controlled motor project successfully achieved the core objective of regulating motor speed based on temperature inputs. Despite challenges with the motor timer feature, the project showcases effective integration of components and adherence to the specified requirements. Future improvements may address the unimplemented features and enhance overall functionality.