

1. Blinking of LED using 8051

microcontroller Using Proteus.

Aim: TO write an assembly language Program to LED
Blink using 8051

Softwares Required: Proteus software.

Program:

ORG 0000H

JMP start

ORG 0100H

start:

SETB P2.0

CALL DELAY

CLR P2.0

CALL DELAY

Loop P

JMP start

DELAY:

MOV R1, #255

L2:

MOV R2, #255

DJNE R2, L1

DJNE R1, L2

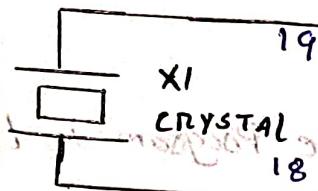
RET

END.

1208 pins QFP 20 pinout

U1

2008 pinout diagram



19 X1A1

18 X1A2

9 RST

2008 pinout diagram

29 PSEN

30 ALE

31 EA

1 PI.0

2 PI.1

3 PI.2

4 PI.3

5 PI.4

6 PI.5

7 PI.6

8 PI.7

AT89C51

2257 115V 0.05A

Procedure

1. Create a New Project

Open Proteus, go to Files New Project name list it, and

Open the Schematic editor

2. Add components.

Click P and search for:
* ATC89C51, LED, crystal oscillator, power (Vcc16VNA)

* Drag and Place them on the workspace.

3. Connect the circuit. (208 pins) (208 pins)
or (208 pins)

4. Load the Program. (208 pins)

5. Run simulation.

Result:

~~Thus the program has been successfully verified and executed.~~

208.89 VTH = 96
VADJ, HADJ
HADJ = 89 VOL
XADJ, SADJ
90 98.02
diff. 98 VOL = XADJ
208.89 VOL = 14
SH. 98 VOL = 54

Program is completed.

2. LED Switching using 8051

Using Proteus.

Aim: To write an assembly language program to perform I/O switching using BIOS.

Software Required: Proteus Required.

Programs: Business

ମୋ କାହିଁଏବେ ମୁଖ୍ୟମାନେ ଯାଏନ୍ତି କଲେ ଅନ୍ଧରୀପ ତମ ଶକ୍ତି
ସମ୍ମଦ୍ଦିଷ୍ଟ

ORG 0000H

UP : MOV P2, #55H

ACALL DELAY

Mov P2, #0AAH

A CALL DELAY

SJMP UP

DELAY : MOV R4, #10

M1 : MOV R3, #255

H_2 : D_{3h} N_2 R_3, H_2

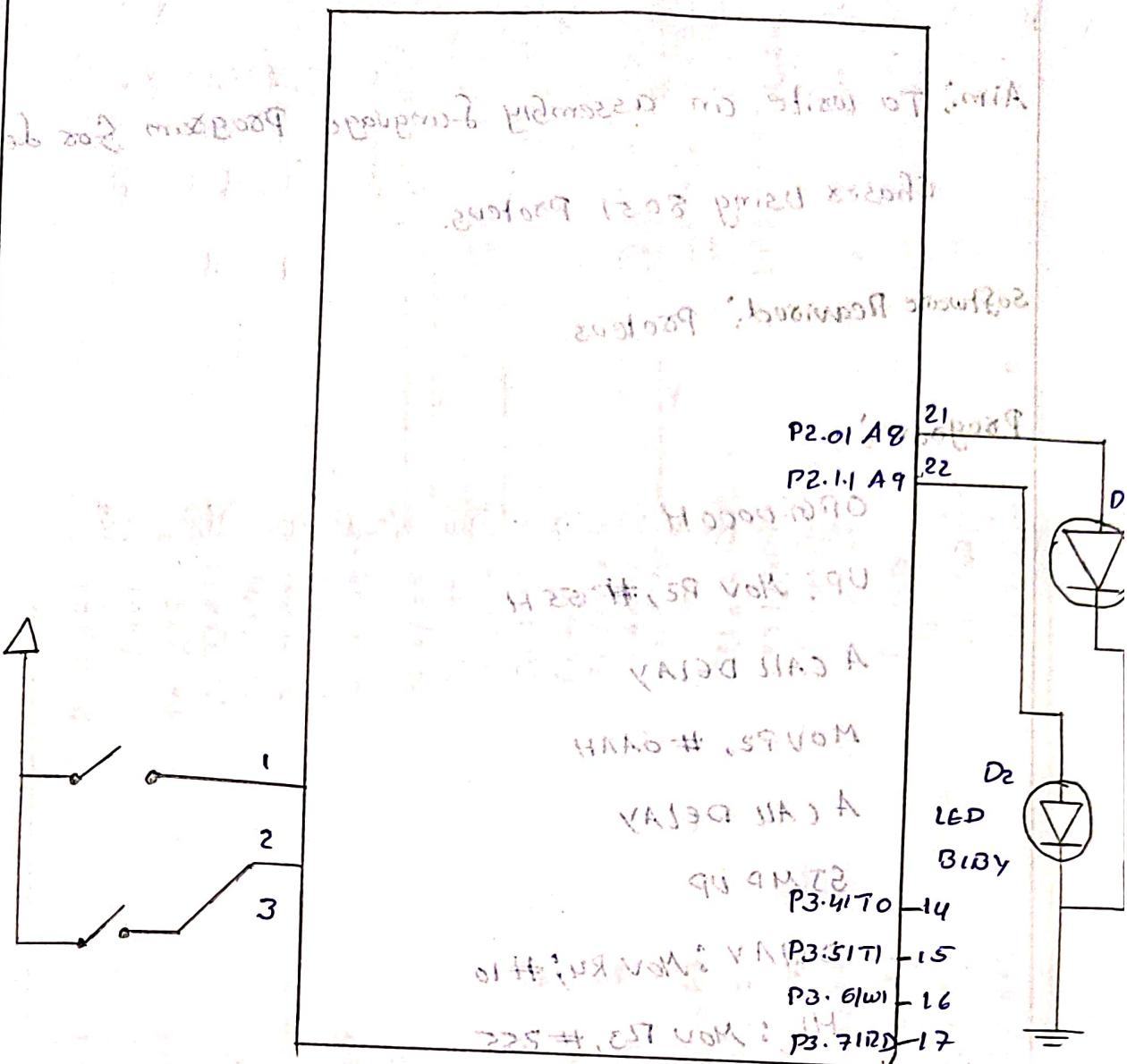
DYN2 R4, H1

RET

END.

8. LED CIRCUIT WITH 3 POSITION SWITCH

10



Result.

~~Thus the program has been successfully verified and executed.~~

3. LED chaser using 8051 Proteus.

Aim: To write an assembly language program for led chaser using 8051 Proteus.

Software Required: Proteus.

Program:

ORG 0000H

UP: MOV R2, #05H

A CALL DELAY

MOV R2, #0AAH

A CALL DELAY

SJMP UP

DELAY: MOV R4, #10

STL R4, L1

L1: MOV R3, #255

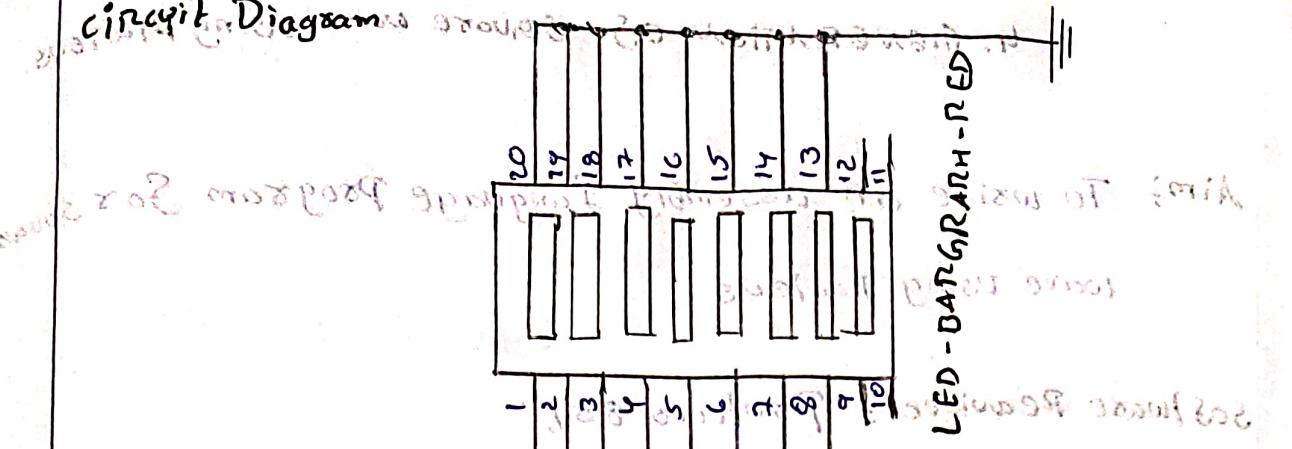
H2: DJNZ R3, H2

DJNZ R4, H,

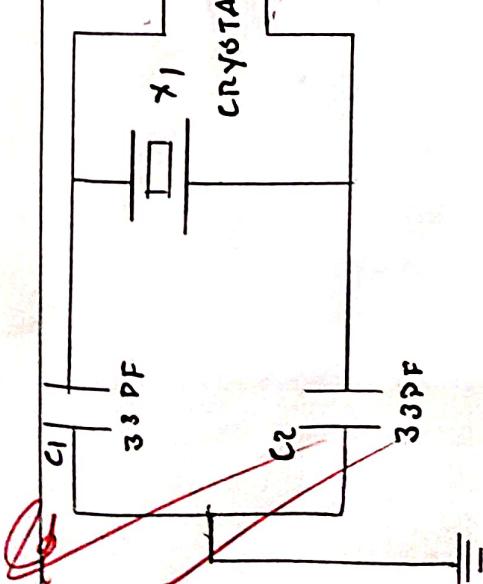
RET

bra begin END

Test



33PF	C1	X1	19	XTAL1	18	XTAL2	9	R5T	29	PSEN	30	ALG	31	GA	1	P1.0/T2	2	P1.1/T2EX	3	P1.2/ECI	4	P1.3/CEX0	5	P1.4/CEX1	6	P1.5/CEX2	7	P1.6/CEX3	8	P1.7/CEX4	
															39	P0.0/AD0	38	P0.1/AD1	37	P0.2/AD2	36	P0.3/AD3	35	P0.4/AD4	34	P0.5/AD5	33	P0.6/AD6	32	P0.7/AD7	21
															22	P2.0/A8	23	P2.1/A9	24	P2.2/A10	25	P2.3/A11	26	P2.4/A12	27	P2.5/A13	28	P2.6/A14	29	P2.7/A15	30
															8	P3.0/RXD	9	P3.1/TXD	10	P3.2/FWD	11	P3.3/INR1	12	P3.4/T0	13	P3.5/T1	14	P3.6/WR	15	P3.7/RD	16
															7	P3.0/RXD	8	P3.1/TXD	9	P3.2/FWD	10	P3.3/INR1	11	P3.4/T0	12	P3.5/T1	13	P3.6/WR	14	P3.7/RD	15



Result: Thus the program has been successfully verified and executed.

4. GENERATION OF SQUARE WAVE USING PROTEUS.

Aim: To write an assembly language program for square wave using Proteus.

Software Required: Proteus 8.5i.

Program:

ORG 0000H

UP: SETB P2.0

A CALL DELAY

CLR P2.0

A CALL DELAY

SJMP UP

DELAY: MOV R4, #35

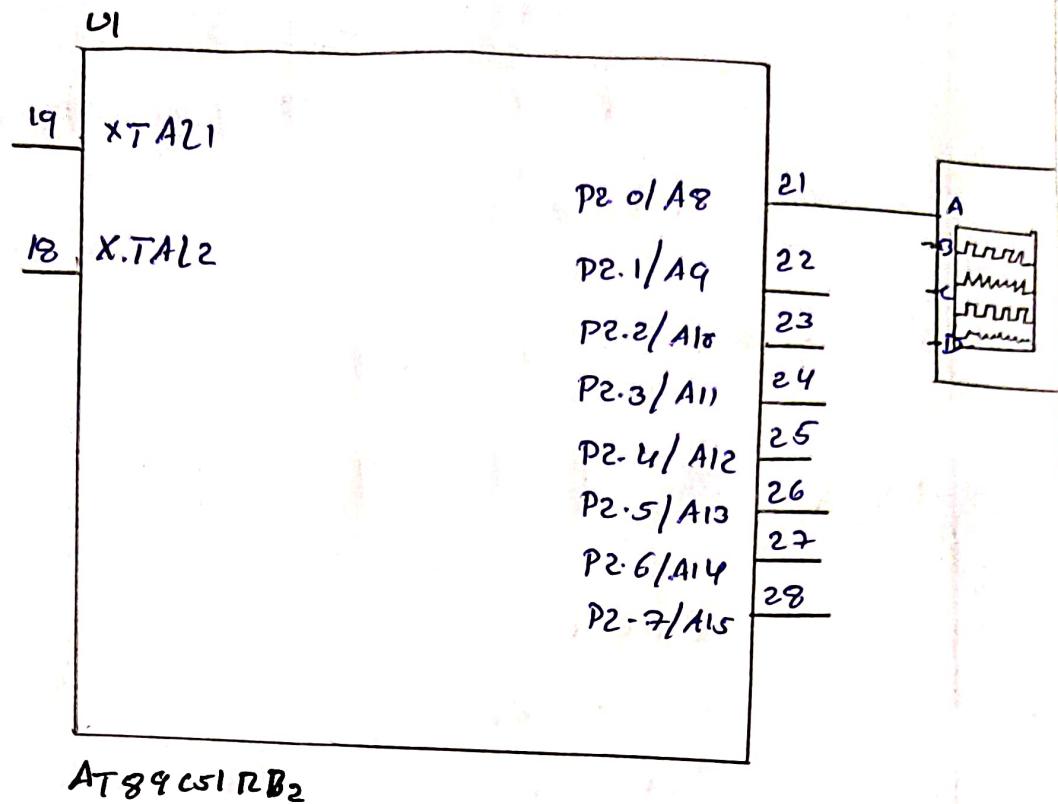
H1: MOV R3, #255

H2: DJNZ R3, H2

DJNZ R4, H1

RET

END.



AT89C51RB2

Result:

~~✓ Thus, the code is executed successfully with no errors for servos wave using Proteus 8051.~~

S. Generation of Triangular Wave

Using Proteus.

Ques: To write an assembly language program for square

wave using Probes

Software required: Delphi 8.05

Program.

ORG COH
NOV 1, #00H
NOV-A, #00H

Nov 20, 2001

INC A
MURP, A

A CALL DELAY
CTMEA, #OF EH

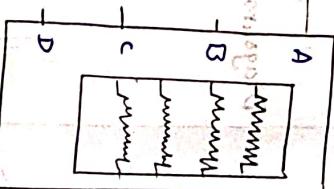
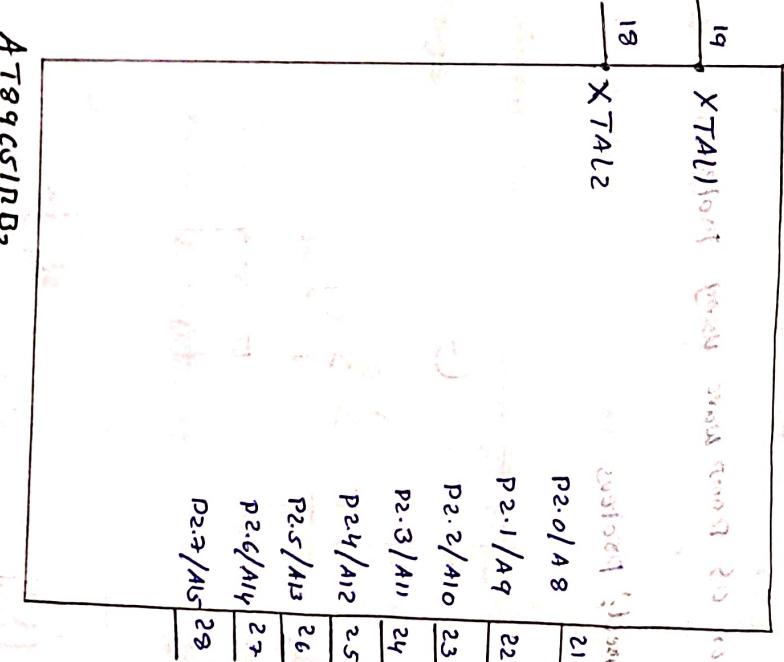
DECA

A CALL

~~CJNEA, #00FFH UPWARD~~

END.

~~Result:~~
Thus the code is executed successfully for languages



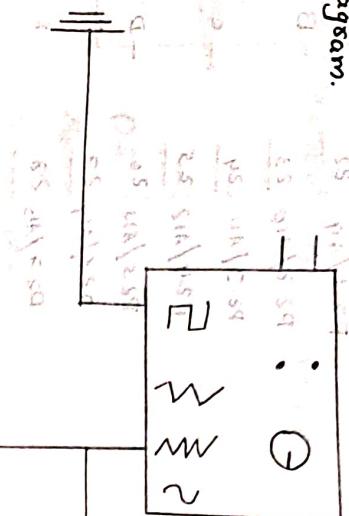
6. Generation of Ramp wave using Proteus

Aim: Generation of Ramp wave using Proteus.

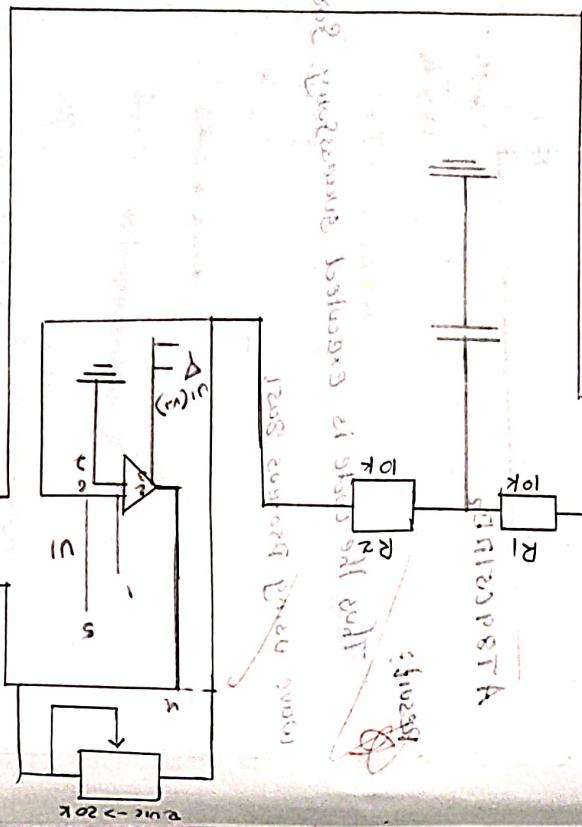
Software Required: Proteus



Diagram:



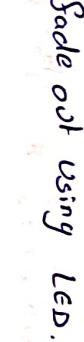
Result: The output is generated successfully for ramp wave.



Proteus

Aim: To write an assembly language program for fade in fade out using LED.

Software Required: Arduino uno & Proteus



Program:

```
int LEDs[3] = {13, 12, 13};
```

```
void setup () {
```

```
for (int i = 0; i < 5; i++) {
```

```
pinMode (LEDs[i], OUTPUT);
```

```
void loop () {
```

```
for (int i = 0; i < 6; i++) {
```

```
digitalWrite (13, HIGH);
```

```
delay (100);
```

```
digitalWrite (13, LOW);
```

```
delay (100);
```

```
for (int i = 0; i < 3; i++) {
```

```
digitalWrite (12, HIGH);
```

```
delay (100);
```

```
digitalWrite (12, LOW);
```

```
for (int i = 0; i < 3; i++) {
```

```
digitalWrite (13, HIGH);
```

```
delay (100);
```

```
digitalWrite (13, LOW);
```

delayed *anulus* *gap* *is* *too* *short* *as* *short* *as*

Procedure:

`digitalWrite(11, LOW)`

Delay (bo):

Circuit Diagrams

2021年1月25日 朝日新聞社編集委員会

17

Thus, the program is executed successfully by Sade in
and Sade out of ignorance.

* Vedic Sanskrit, the original of all Indian literature.

and the other two were lost.

THE CLOTHESLINE 5

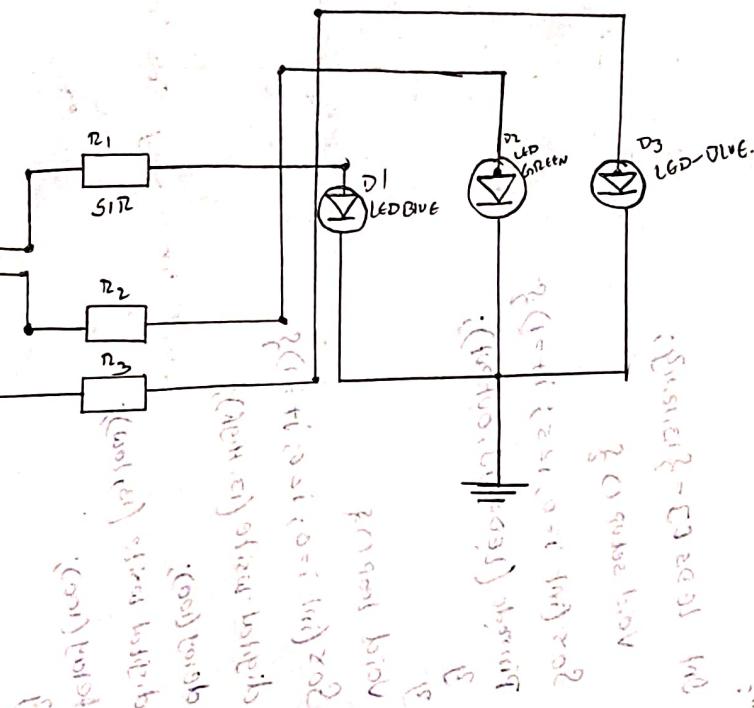
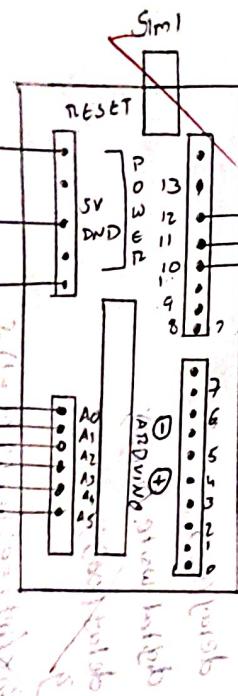
Revised edition 1965
Original edition 1963

4 CEDAR VALLEY COLLEGE

卷之三

1950-1952 1953-1954 1955-1956 1957-1958

卷之三



* connect the other END of LED to Ground (GND) with
* connect the GND pins to Arduino Uno.

8. Clockwise Rotation of stepper motor using 8051 using Proteus.

Objectives: To observe the rotation of stepper motor clockwise.

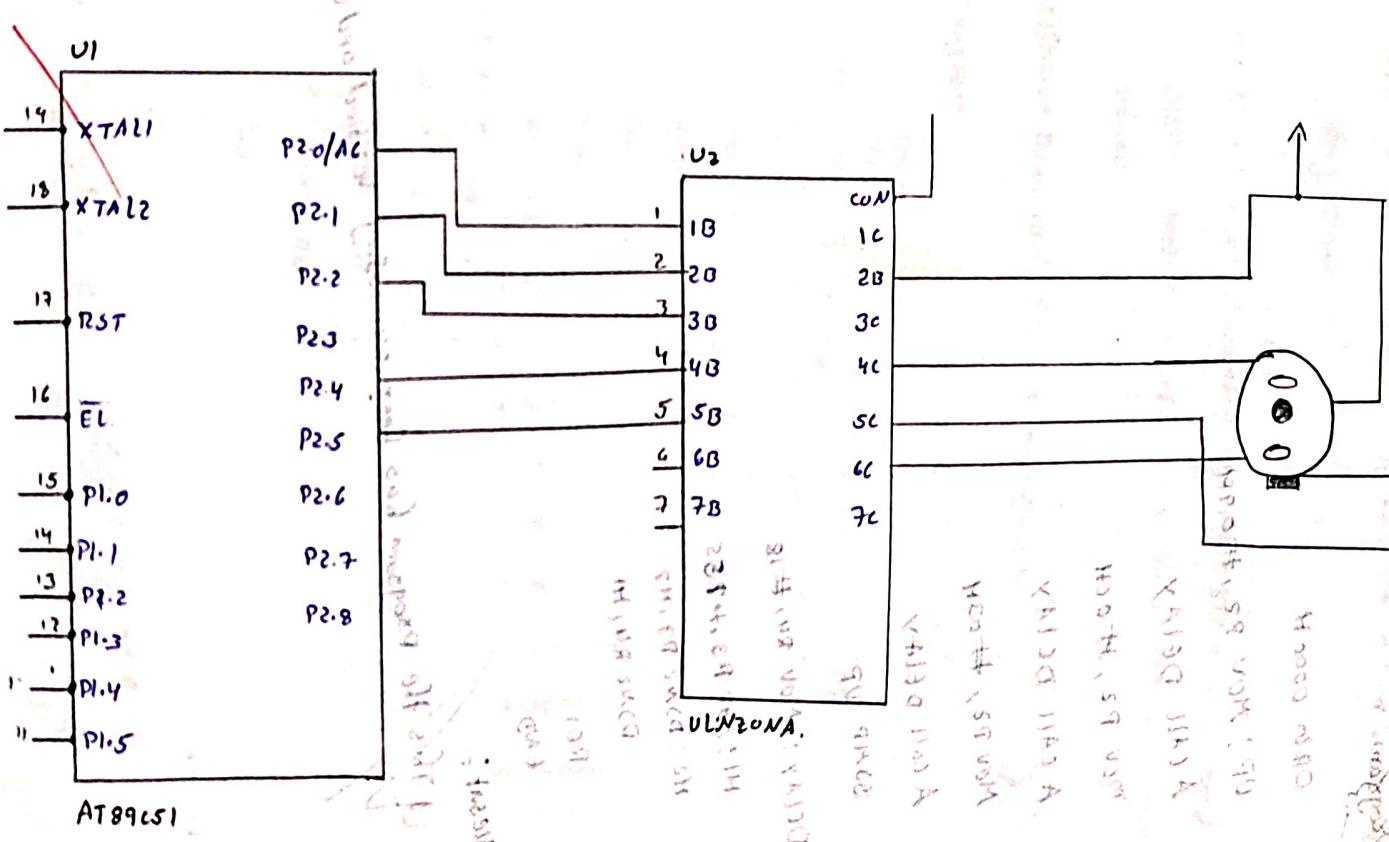
Aim: To observe the rotation of stepper motor clockwise.

Procedure:

* Create a new project, open Proteus goto File > New Project, name it, and open the schematic editor.

* Add components to the schematic editor now click **E** and - AT89CS1, Battery, Dutton, motor Stepper and ULN2003A connect the stepper motor to the battery and stepper motor to the ULN2003A

* connect the bottom to the pin P3.10 INXO.
Input:
ULN2003A - P2.0, P2.1, P2.2, P2.3
Stepper motor - 16, 15, 14, 12
Button - P3.0.
Output:
Stepper motor connected to the battery



Program

O12n 0000H

UP: MOV P2, #09H

A CALL DELAY

MOV P2, #0CH

A CALL DELAY

MOV P2, #03H

A CALL DELAY

SJMP UP

DELAY: MOV R4, #18

M1: MOV R3, #285

M2: DJNZ R3, M2

DJNZ R4, M1

RET

END.

Result:

~~Thus the Program has been successfully verified and exec~~

Q. Anticlock wise Rotation of stepper motor using 8051 using Proteus.

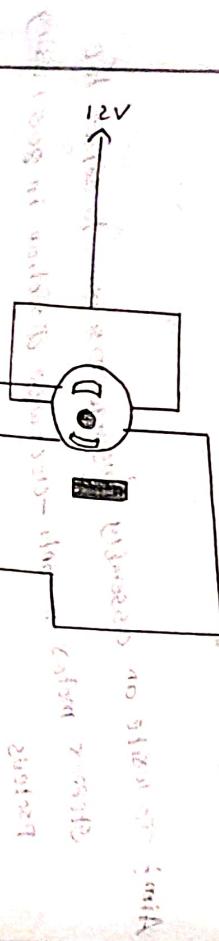
Aim: To write an assembly language program to rotate the stepper motor in anti-clockwise direction in 8051 using Proteus.

Software Required: Proteus 8 software

Program:

```
ORG 0000H
UP: MOV P2, #03H
    ACALL DELAY
    MOV P2, #06H
    ACALL DELAY
    MOV P2, #09H
    ACALL DELAY
    SJMP UP
DELAY: MOV R4, #18
    H1: MOV R3, #255
    H2: DJNZ R3, H2
    DJNZ R4, H1
    RET
END.
```

10. Rotating Stepper motor 90° Angle in 8051 using Proteus.



Aim: To write an assembly language program rotating the Stepper motor 90° in 8051.

Software Required: Proteus 8051

Program:

ORG 0000H

MAIN:

Mov P2, #00H

Mov P1, #0FH

Mov P2, #06H

CALL DELAY

SJMP

DELAY:

Mov R0, #50

DELAY - Loop1

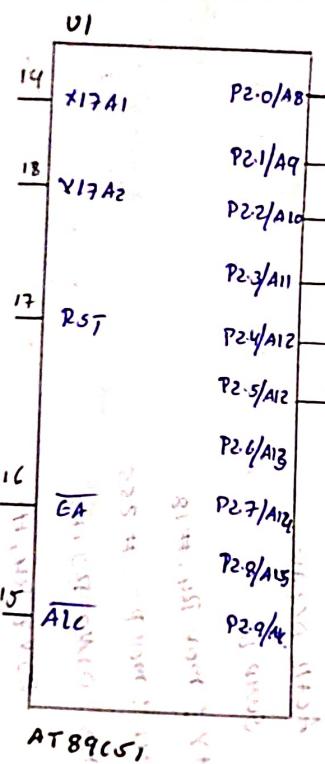
Mov R1, #127

DELAY - Loop2

Mov R2, R1, DELAY - Loop2

MOV R2, R1, DELAY - Loop1

END



Output:

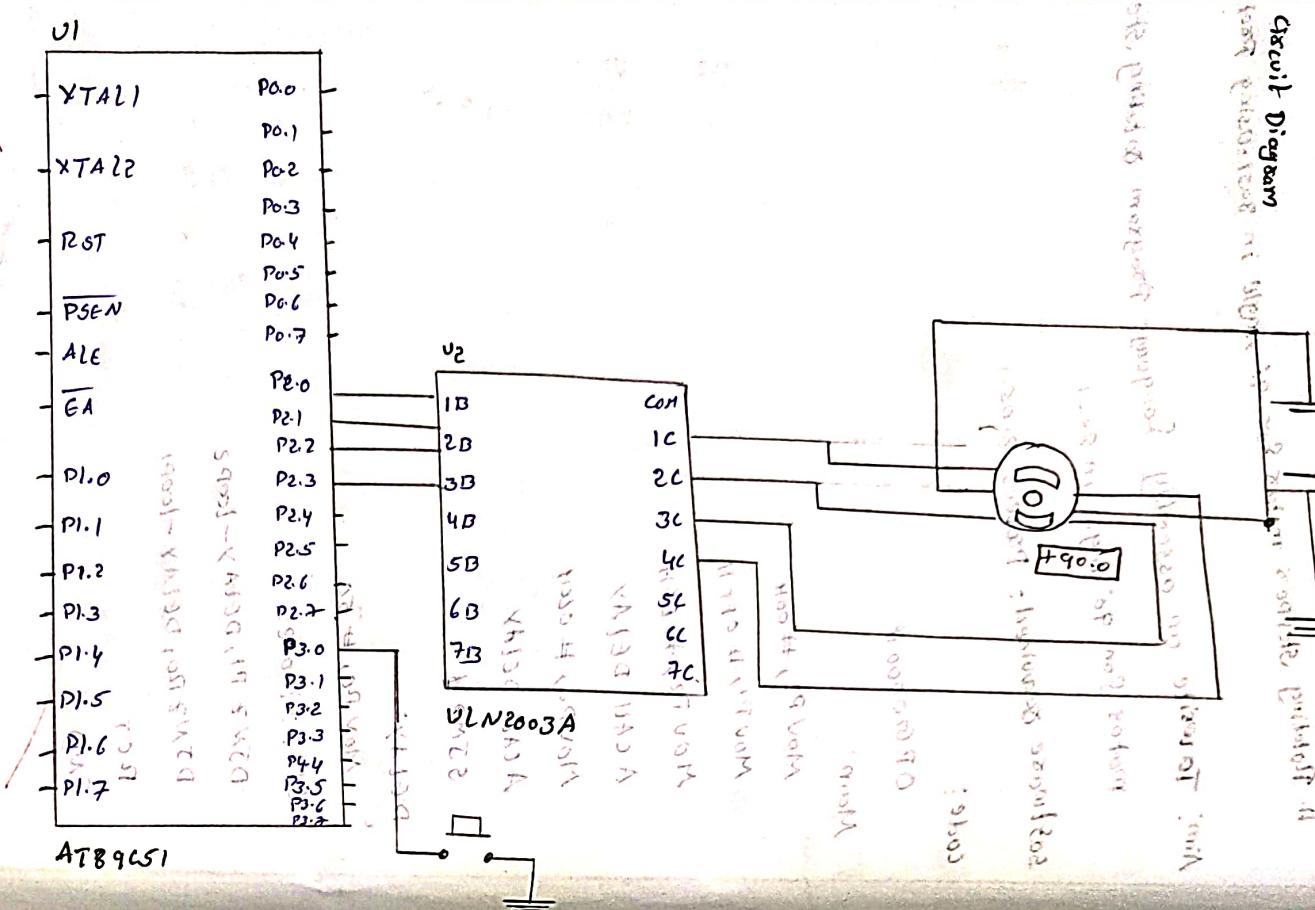
The stepper motor is rotating in anticlockwise direction.

~~Result:~~

~~Thus the program has been successfully revised and executed.~~

Circuit Diagram

Block diagram showing a 24VDC motor connected to a relay coil. The relay coil is controlled by a microcontroller output. A 10kΩ resistor is connected in series with the coil. The microcontroller also provides power to the motor via a diode and a 1N4007 diode.



Aim: To write an assembly language program to interface relay and bulb with Proteus.

Software Required: Proteus & software.

Program.

076 000H

UP:SET P2.0

A CALL DELAY

CLR P2.0

ACALL DELAY

SJMP UP

DELAY: MOVR4, #18

M1: MOVI #3, #255

H2: DJNZ R4, #2

DJNZ R4, H1

RET

END

~~Result:~~ Thus the program is executed successfully for 90 degree is rotating stepper motor.

13. Interfacing of Relay and LED with 8051 using Proteus

Protues

Aim: To write an assembly language to interface relay

LED with 8051 using proteus.

Software Required: Proteus and software.

Code:

ORG 0000H

MOV P1, #00H

Main - loop:

SETB P1.0;

A CALL DELAY

CLR P1.0

GJMP Main Loop

DELAY:

Mov R1, #255

Delay 1:

Mov R2, #255

Delay 2:

DJMN2 R2, Delay2

DJMN2 R1, Delay1

Delay1: mov R1, #255

Delay2: mov R2, #255

Delay3: mov R3, #255

Delay4: mov R4, #255

Delay5: mov R5, #255

Delay6: mov R6, #255

Delay7: mov R7, #255

Delay8: mov R8, #255

Delay9: mov R9, #255

Delay10: mov R10, #255

Delay11: mov R11, #255

Delay12: mov R12, #255

Delay13: mov R13, #255

Delay14: mov R14, #255

Delay15: mov R15, #255

Delay16: mov R16, #255

Delay17: mov R17, #255

Delay18: mov R18, #255

Delay19: mov R19, #255

Delay20: mov R20, #255

Delay21: mov R21, #255

Delay22: mov R22, #255

Delay23: mov R23, #255

Delay24: mov R24, #255

Delay25: mov R25, #255

Delay26: mov R26, #255

Delay27: mov R27, #255

Delay28: mov R28, #255

Delay29: mov R29, #255

Delay30: mov R30, #255

Delay31: mov R31, #255

Delay32: mov R32, #255

Delay33: mov R33, #255

Delay34: mov R34, #255

Delay35: mov R35, #255

Delay36: mov R36, #255

Delay37: mov R37, #255

Delay38: mov R38, #255

Delay39: mov R39, #255

Delay40: mov R40, #255

Delay41: mov R41, #255

Delay42: mov R42, #255

Delay43: mov R43, #255

Delay44: mov R44, #255

Delay45: mov R45, #255

Delay46: mov R46, #255

Delay47: mov R47, #255

Delay48: mov R48, #255

Delay49: mov R49, #255

Delay50: mov R50, #255

Delay51: mov R51, #255

Delay52: mov R52, #255

Delay53: mov R53, #255

Delay54: mov R54, #255

Delay55: mov R55, #255

Delay56: mov R56, #255

Delay57: mov R57, #255

Delay58: mov R58, #255

Delay59: mov R59, #255

Delay60: mov R60, #255

Delay61: mov R61, #255

Delay62: mov R62, #255

Delay63: mov R63, #255

Delay64: mov R64, #255

Delay65: mov R65, #255

Delay66: mov R66, #255

Delay67: mov R67, #255

Delay68: mov R68, #255

Delay69: mov R69, #255

Delay70: mov R70, #255

Delay71: mov R71, #255

Delay72: mov R72, #255

Delay73: mov R73, #255

Delay74: mov R74, #255

Delay75: mov R75, #255

Delay76: mov R76, #255

Delay77: mov R77, #255

Delay78: mov R78, #255

Delay79: mov R79, #255

Delay80: mov R80, #255

Delay81: mov R81, #255

Delay82: mov R82, #255

Delay83: mov R83, #255

Delay84: mov R84, #255

Delay85: mov R85, #255

Delay86: mov R86, #255

Delay87: mov R87, #255

Delay88: mov R88, #255

Delay89: mov R89, #255

Delay90: mov R90, #255

Delay91: mov R91, #255

Delay92: mov R92, #255

Delay93: mov R93, #255

Delay94: mov R94, #255

Delay95: mov R95, #255

Delay96: mov R96, #255

Delay97: mov R97, #255

Delay98: mov R98, #255

Delay99: mov R99, #255

Delay100: mov R100, #255

Delay101: mov R101, #255

Delay102: mov R102, #255

Delay103: mov R103, #255

Delay104: mov R104, #255

Delay105: mov R105, #255

Delay106: mov R106, #255

Delay107: mov R107, #255

Delay108: mov R108, #255

Delay109: mov R109, #255

Delay110: mov R110, #255

Delay111: mov R111, #255

Delay112: mov R112, #255

Delay113: mov R113, #255

Delay114: mov R114, #255

Delay115: mov R115, #255

Delay116: mov R116, #255

Delay117: mov R117, #255

Delay118: mov R118, #255

Delay119: mov R119, #255

Delay120: mov R120, #255

Delay121: mov R121, #255

Delay122: mov R122, #255

Delay123: mov R123, #255

Delay124: mov R124, #255

Delay125: mov R125, #255

Delay126: mov R126, #255

Delay127: mov R127, #255

Delay128: mov R128, #255

Delay129: mov R129, #255

Delay130: mov R130, #255

Delay131: mov R131, #255

Delay132: mov R132, #255

Delay133: mov R133, #255

Delay134: mov R134, #255

Delay135: mov R135, #255

Delay136: mov R136, #255

Delay137: mov R137, #255

Delay138: mov R138, #255

Delay139: mov R139, #255

Delay140: mov R140, #255

Delay141: mov R141, #255

Delay142: mov R142, #255

Delay143: mov R143, #255

Delay144: mov R144, #255

Delay145: mov R145, #255

Delay146: mov R146, #255

Delay147: mov R147, #255

Delay148: mov R148, #255

Delay149: mov R149, #255

Delay150: mov R150, #255

Delay151: mov R151, #255

Delay152: mov R152, #255

Delay153: mov R153, #255

Delay154: mov R154, #255

Delay155: mov R155, #255

Delay156: mov R156, #255

Delay157: mov R157, #255

Delay158: mov R158, #255

Delay159: mov R159, #255

Delay160: mov R160, #255

Delay161: mov R161, #255

Delay162: mov R162, #255

Delay163: mov R163, #255

Delay164: mov R164, #255

Delay165: mov R165, #255

Delay166: mov R166, #255

Delay167: mov R167, #255

Delay168: mov R168, #255

Delay169: mov R169, #255

Delay170: mov R170, #255

Delay171: mov R171, #255

Delay172: mov R172, #255

Delay173: mov R173, #255

Delay174: mov R174, #255

Delay175: mov R175, #255

Delay176: mov R176, #255

Delay177: mov R177, #255

Delay178: mov R178, #255

Delay179: mov R179, #255

Delay180: mov R180, #255

Delay181: mov R181, #255

Delay182: mov R182, #255

Delay183: mov R183, #255

Delay184: mov R184, #255

Delay185: mov R185, #255

Delay186: mov R186, #255

Delay187: mov R187, #255

Delay188: mov R188, #255

Delay189: mov R189, #255

Delay190: mov R190, #255

Delay191: mov R191, #255

Delay192: mov R192, #255

Delay193: mov R193, #255

Delay194: mov R194, #255

Delay195: mov R195, #255

Delay196: mov R196, #255

Delay197: mov R197, #255

Delay198: mov R198, #255

Delay199: mov R199, #255

Delay200: mov R200, #255

Delay201: mov R201, #255

Delay202: mov R202, #255

Delay203: mov R203, #255

Delay204: mov R204, #255

Delay205: mov R205, #255

Delay206: mov R206, #255

Delay207: mov R207, #255

Delay208: mov R208, #255

Delay209: mov R209, #255

Delay210: mov R210, #255

Delay211: mov R211, #255

Delay212: mov R212, #255

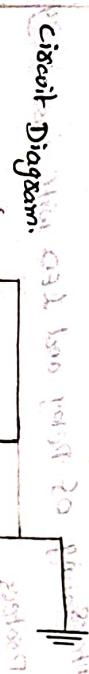
Delay213: mov R213, #255

Delay214: mov R214, #255

Delay215: mov R215, #255

Delay216: mov R216, #255

Circuit Diagram



14. Digital Clock on LCD

Aim: To write an assembly language program to display digital clock on LCD using Proteus.

Software Required: Proteus Software.

Program:

```
ORG 0000H
MOV R7, #00H
MOV R6, #00H
MOV RS, #00H
```

ACALL INT-LCD

MOV A, #0CH

ACALL END-WRITC

ACALL DELAY-SHORT

MVFA, #01H

ACALL CMD-WRITE

ACALL DELAY-SHORT

MVFA, #01H

ACALL CMD-WRTE

ACALL DELAY-SHORT

RET

DELAY-1-SEC: MOV R3, #00H
MOV R2, #00H
MOV R1, #00H

DELAY-loop

MOV R1, #255

DELAY-loop-INNER

Output:
The LED connected through the delay will blink with a controlled on and off duration.
Result:

This program has been successfully tested and executed.

15. ADC interfacing with pot and LED Display using Proteus.

poorus.

Aim: Write an assembly language program for ADC interfacing with 8051 and LED Display using Proteus.

Sophomore reviewed: Keil, Prokow

code:

ON 60000 H

NOV 11, 1964

SET P3.0

A CALL DELAY

CLN R3.0

SET P3

Nov A, 7

65

A CALL DELAY

SMP Main

卷之三

100100, #250

D J N 2 B. 13

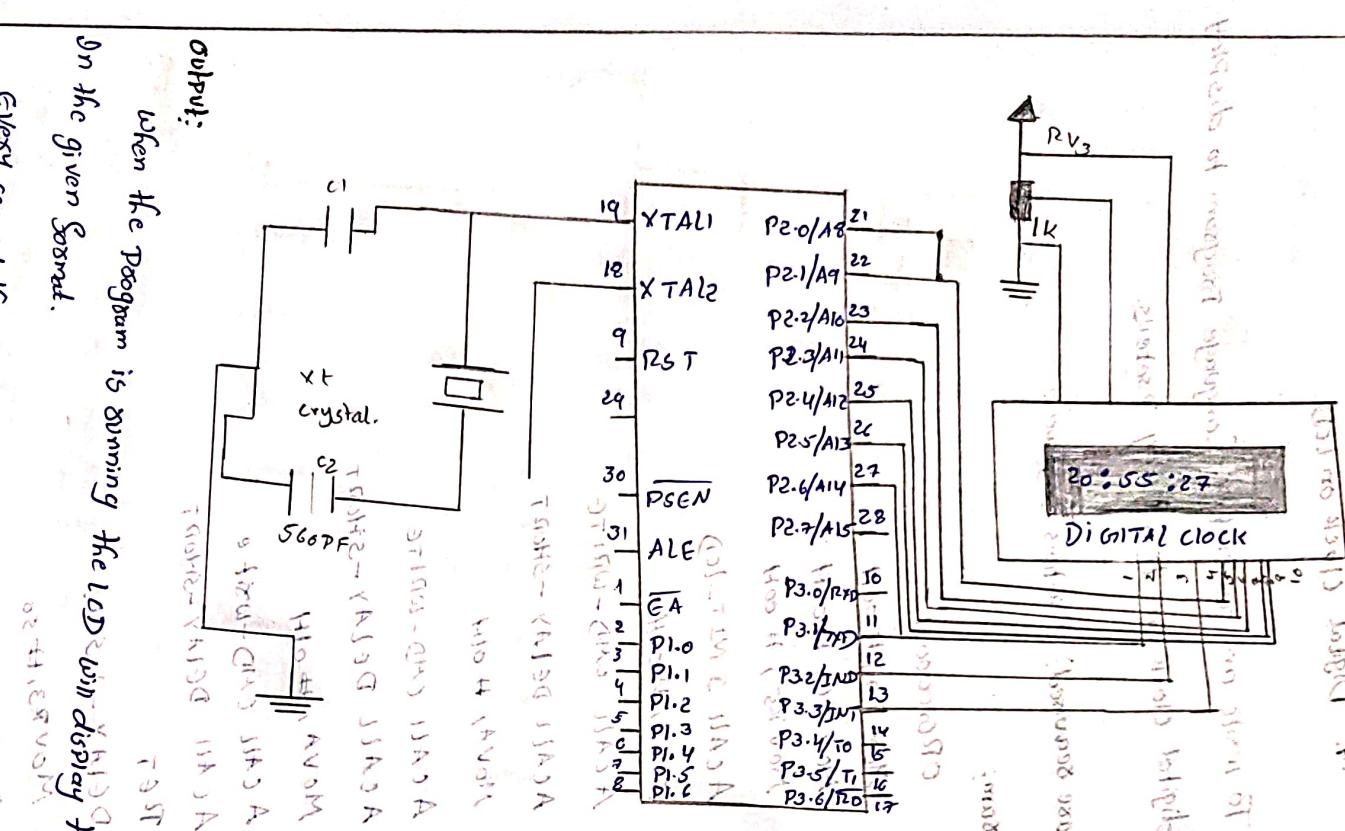
JUN 2 2010 DELAY

245

37

12.

三



When the program is running the LCD will display the time
In the given format.

Output: 

Every second the display will show : 9005 - 9005 0

Result: Increment the score by

Thus the assumption is

LCD with using 386AD5000E program to display digital clock on

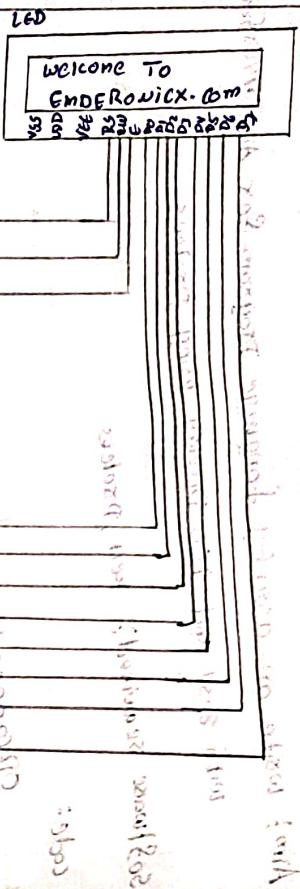
16 ADC Interfacing with Basys and LCD Display using prokuec

Aim: To write an assembly language program for ADC, the interfacing with 8051 and LCD display using Proteus.

Software Required: Prolog & software

Program

۲۰



0R2G 0000 H

100P

SETBSC

A CAN DELAY

clr ALE

DELAYE

MOUNTAIN, #250

D J N 2 No. 1

卷之三

RECEIVED
LIBRARY OF THE
UNIVERSITY OF TORONTO LIBRARIES
1962

三

When the analog input voltage is varied the ADC converts it into digital value.

100

when the analogy

value is displayed on LED.

thus the assembly language program is created in !

14. Interfacing of relay and DC Motors with Proteus
8051.

R. L. B. 128

Aim: To write an assembly language program for interfacing of relay and DC Motors with Proteus 8.051.

Software reviewed: Kecil 11 version, Proteus 8051.

Program:

100000

Hoot, Hoot

Nov 1, 2011

J B Acc. 01

STAMP TURN

b2 v
b3 a
b3 b
b3 c

3678 RELAY

卷之三

卷之三

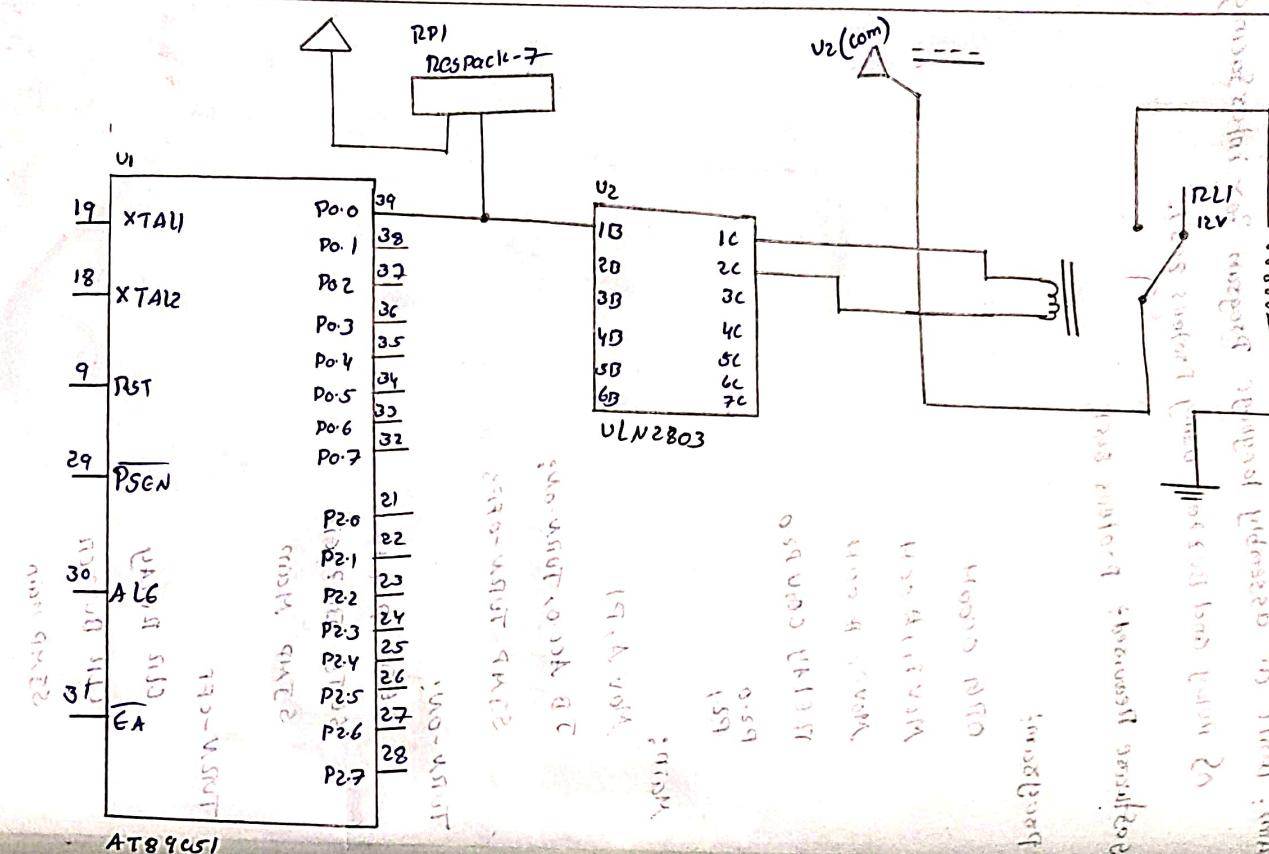
CIR RELAY

CLR MOTOR

SJM Man

-G-

卷之三



Result.

A language program for interfacing of relay and driver is executed successfully and verified.

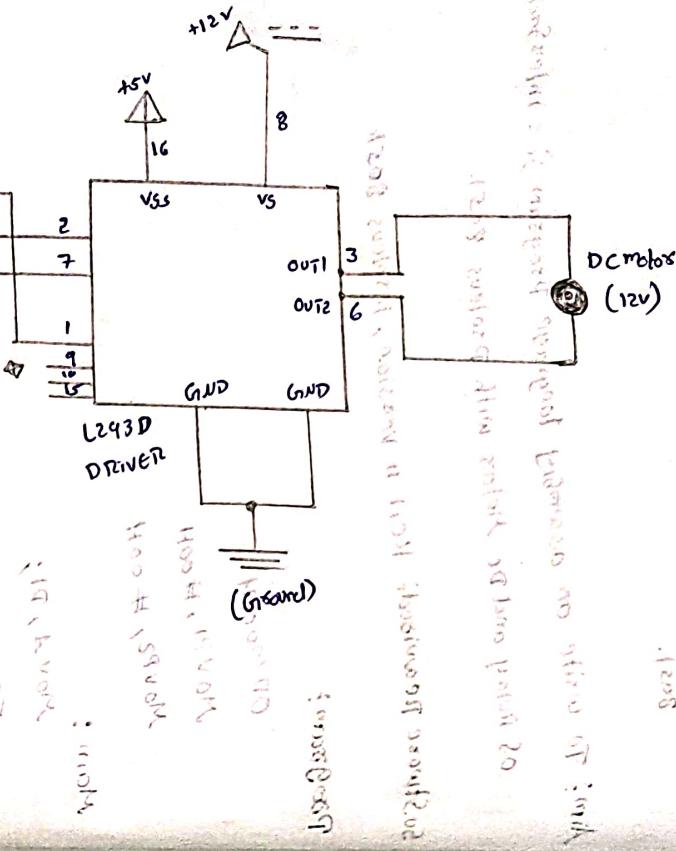
20. Display counting of Numbers on LCD using Pseudo.

Consequently it is necessary to provide an alternative link

• write an assembly language program
• writing of numbers on LCD using Proteus

1. *Leucosia* *leucostoma* *leucostoma* *leucostoma*

Sophomore Reward: Proverbs 30:5



Program:

ORG 0000H
MOV DPTR

ACAH 6MD-1181b

Nov A

mai, y! loop!

卷之三

卷一百一十一

Nov 4, No. 10
1953

卷之三

卷之三

DELAY: *vif*
bel
bco
bcc
bcr
bcl
bcs
bct
bca
bcb
bcs
bce
bcd
bcb
bca
bct
bcc
bcc

DELAWARE

DTN² B1 2011

DIN 2

Results

Thus the code is verified and

21. Display A Number in 7 segment Display with 8051 Proteus

Aim: To write an assembly language program for displaying a number in 7 segment display.

Software Required: Proteus 8.051

Code:

```
ORG 00000H
MOV R1, #92H
```

Here

SJMP HERE; infinite loop to keep the display active

Delay:

```
MOV R2,
```

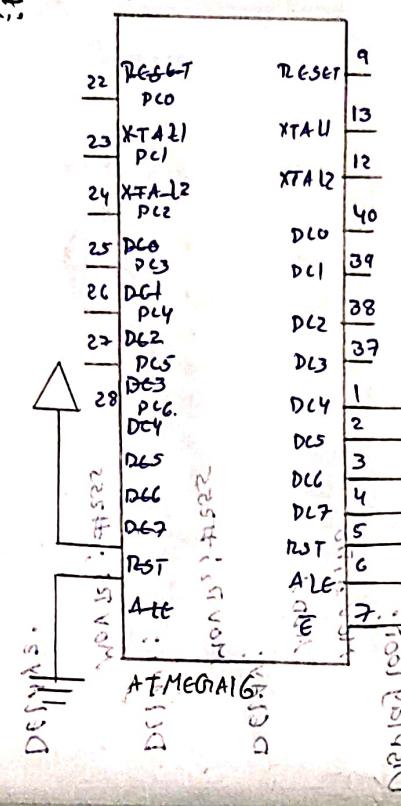
```
MV R1,
```

```
DJNZ R1, L1
```

```
L1:
```

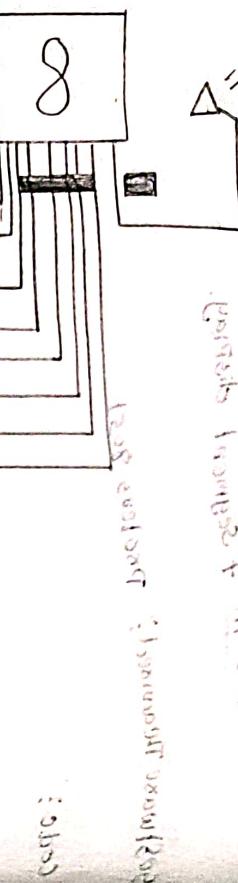
```
RET
```

```
END
```



22. Counting up the numbers in 7-segment Display with 8051 using Proteus

Aim: To write an assembly language program for counting up the numbers in 7-segment Display.



Software Required: Proteus 8as1

Program:

ORG 0000H

Mov R0,

Mov R4,

CALL Display

Delay Subroutine

Delay:

Mov R2, #255

Delay 1:

Mov R1, #255

Delay 2:

DJNZ R2, Delay2

DJNZ R1, Delay1

RET

END.

This corresponds to the digits.
Load the Hexafile and output will be displayed.

Result:
Thus, the program is executed successfully.

23. Display a Hexadecimal number in 7 segment display using 8051

Aim: To write an assembly language program to display number in 7 segment display using Proteus.

Software Required: Proteus 8051.

Program.

```
ORG 0000H
MOV R0, #00H
MOV A, R0
ACALL Display
A CALL Delay
INC R0
CJNE R0, #00H, Delay
MOV R2, #255
Delay1:
    MOV R1, #255
    Delay2:
        MOV R0, #0FFH
        SJMP Delay2
    RET
```



Output:

- * The 7-segment display will sequentially show numbers from 0 to 9, one at a time.
- * The sequence will repeat in an infinite loop, with display numbers

Result:

Thus the output is created successfully and visible.

Program in assembly language to print a message

Message 1208 from printer

24. Study of PCB printing using EagleCAD.

Aim: To write an assembly language program for study of PCB printing using eaglecad.

Software Required: Eagle-CAD, KiCad

Procedure:

1. Schematic Creation:

- * Open Eagle CAD and create a new project.

* Add components from the library, connect them using wires, and assign values.

2. PCB Layout:

* Switch to PCB layout view

* Arrange components and create traces between them

* Use Auto-Solder Sox Solder results.

3. Export PCB for Printing

* Go to File > CAM Processor, select the Gerber format

and generate the file for PCB fabrication.

Output:

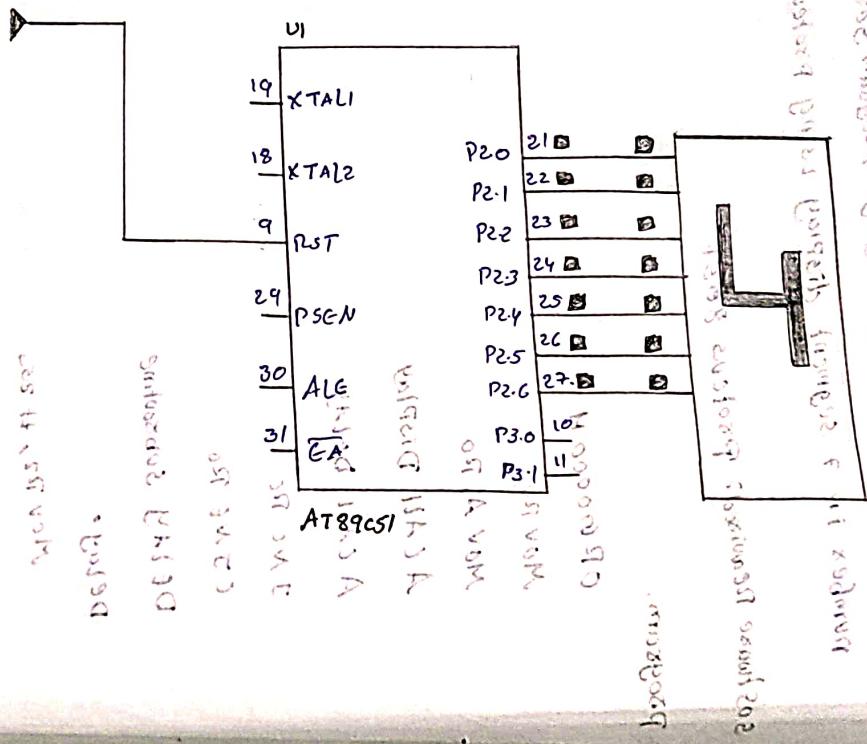
The 7-segment display will sequentially show hexadecimal numbers: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. Once

Result:

Thus the output is verified and executed successfully.

Result:

Thus the output is verified and executed successfully.



25. Blinking of LED using Arduino uno

Aim: To link an led connected to an arduino uno board at regular intervals.

Result: LED blinks at regular intervals.

Apparatus:

- * Arduino uno
- * LED
- * Resistors
- * Jumper wires
- * Bread board

Procedure:

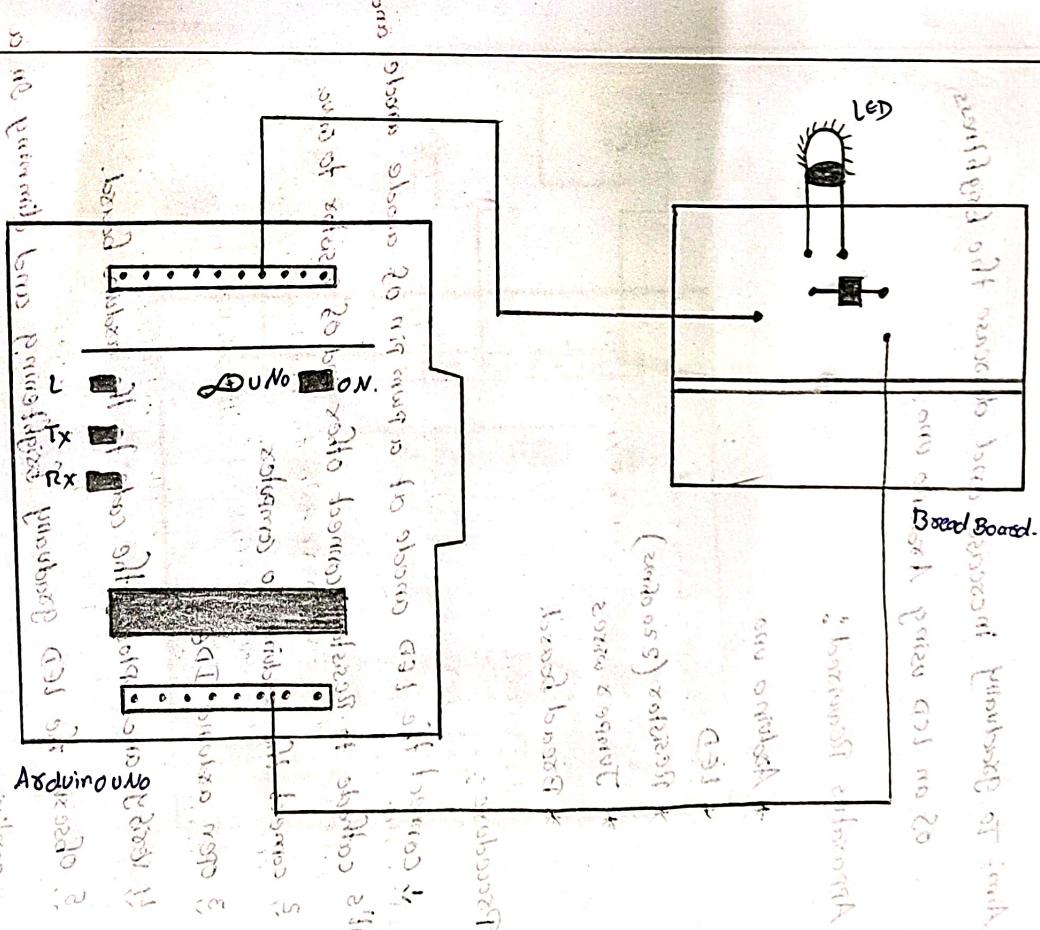
1. Connect the LED's longer lead (Anode) to digital pin and the shorter lead (cathode) to digital pin.
2. Connect the Arduino to a computer using USB cable.
3. Open the Arduino IDE.
4. Verify and upload the code to the Arduino Board.
5. Observe the LED blinking at 1-second intervals.

Output:

- * The LED will turn on for 1 second and turn off for 1 second.
- * The LED will turn ON for 1 second and turn OFF for 1 second.
- * The cycle will repeat indefinitely.

Result:

- We successfully completed the project.



26. Fading of an LED using Arduino uno.

Aim: To gradually increase and decrease the brightness of an LED using Arduino uno.

Apparatus Required:

- * Arduino uno
- * LED
- * Resistor (220 ohms)
- * Jumper wires
- * Bread board

Procedure:

1. connect the LED anode or a pwm pin of anode anode and it's cathode to resistor connect other end of resistor to GND.
2. connect the Arduino to computer.

3. open arduino IDE.

4. write and upload the code to the arduino board.

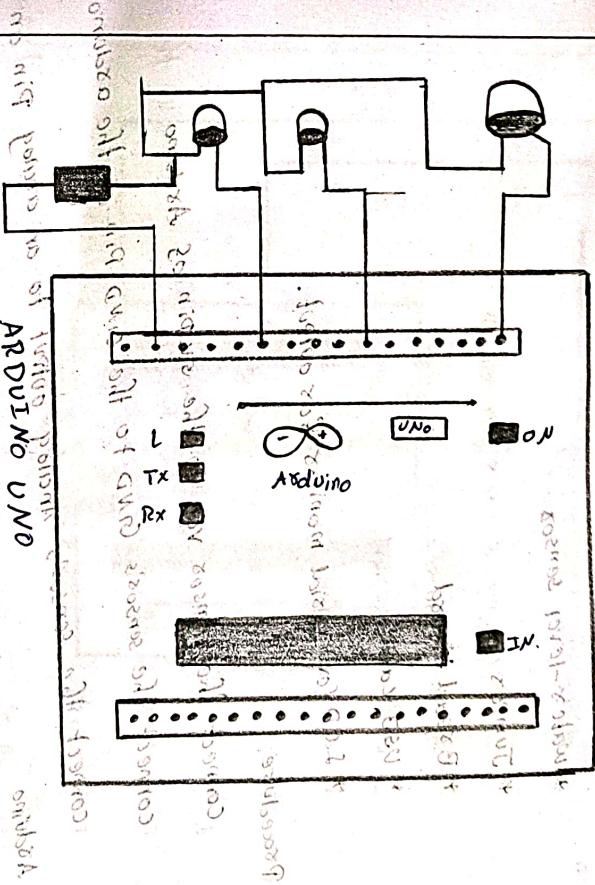
5. observe the LED gradually brightening and dimming in a continuous loop.

about.

The LED brightness gradually increases.

The LED brightness gradually decreases.

The fading process repeats continuously.



Result:

When the above program is executed successfully, the LED will start fading. The LED will gradually increase in brightness until it reaches its maximum brightness, and then it will gradually decrease back to its minimum brightness. This process will repeat continuously.

The LED brightness gradually increases.

The LED brightness gradually decreases.

The fading process repeats continuously.

27. Interfacing a water-level sensor with Arduino Uno.

Aim: To detect and display water levels using a water level sensor connected to Arduino Uno.

Apparatus Required.

- * Arduino uno
- * water-level sensor
- * Jumper
- * Bread board
- * USB cable
- * LCD (or) serial monitor for output.

Procedure.

- Connect the sensor Vcc to the 5V pin of Arduino
- connect the sensor's GND to the GND pin of the Arduino
- connect the sensor's analog output to an analog pin on

Arduino

Open the serial monitor in the Arduino IDE to observe real-time

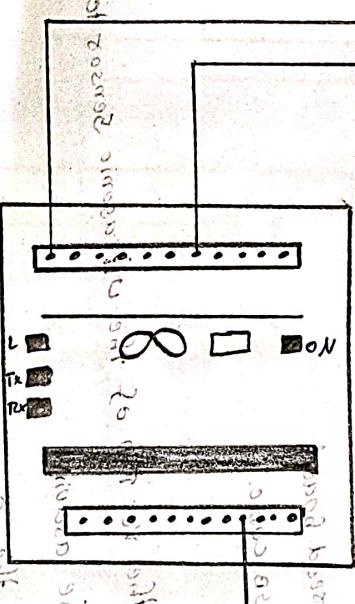
water level-sensor.

Set thresholds to trigger alerts for high (or) low water levels.

~~the sensor detects water levels based on its immersion depth.~~

~~Real-time water-level readings are displayed on the serial monitor.~~

~~Changes in sensor reading denotes variations in water levels.~~



Sensors.

Result:
Thus the above program is executed successfully Water level

~~changes in sensor reading denotes variations in water levels.~~

28. Interfacing an ultrasonic sensor with Arduino Uno

Aim:

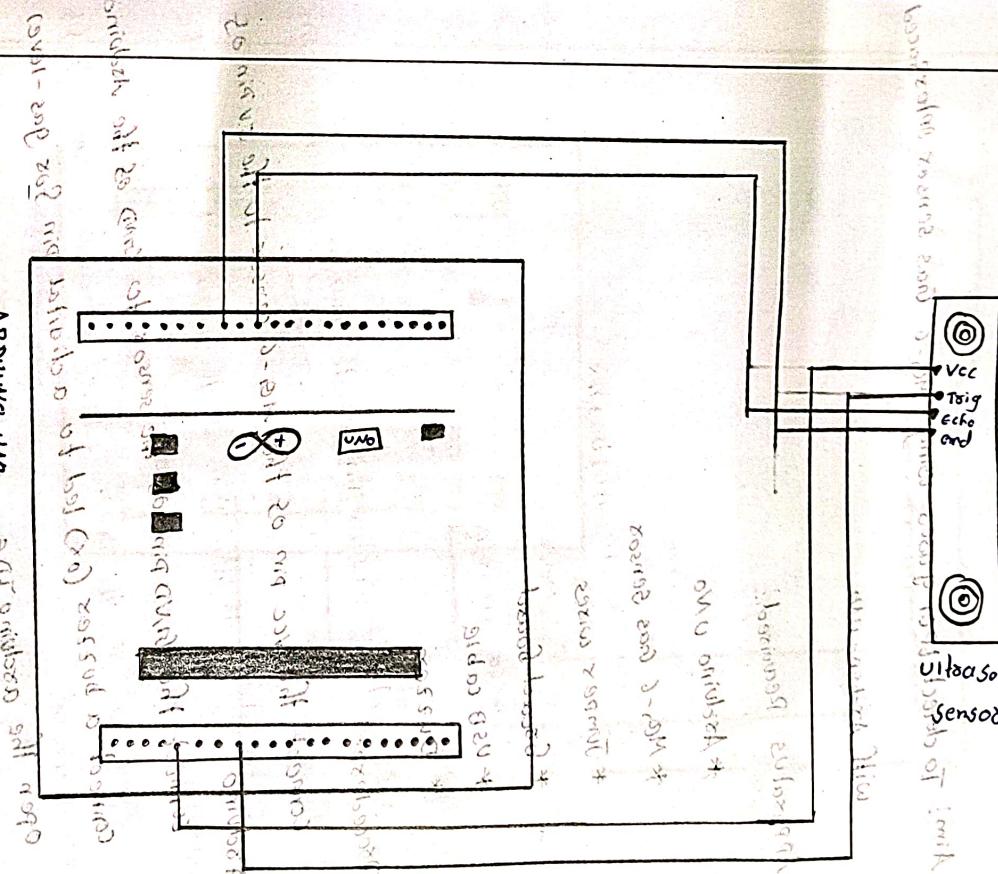
To measure the distance of an object using an ultrasonic sensor (HC-SR04) and Arduino Uno.

Apparatus Required:

- * Arduino Uno Board
- * Ultra sonic sensor (HC-SR04)
- * Jumper wires
- * Bread Board
- * USB cable.

Procedure:

- 1, connect the Vcc pin of the ultrasonic sensor to the 5V pin of the arduino.
- 2, connect the GND pin of the ultrasonic sensor to the GND pin of the arduino.
- 3, connect the Trig pin of the sensor to pin 9 of the Arduino.
- 4, open the arduino.
- 5, open the serial monitor in the Arduino and observe distance readings.
- 6, Display distance reading on a LCD.



Report:

Thus the above program is executed successfully. Ultrasonic sensor emits ultrasonic waves and measures the time taken for them return.

The ultrasonic Sensor emits ultrasonic waves and measures the time.

29. MQ-6 gas sensor Interfacing with arduino uno.

Aim: To detect LPG gases using MQ-6 Gas sensor interfaced with Arduino Uno.

Apparatus Required:

- * MQ-6 Gas Sensor
- * Arduino Uno
- * Jumper wires
- * Bread Board
- * USB cable
- * Buzzer

Procedure:

Connect the Vcc pin of the MQ-6 sensor to the 5V pin of Arduino.

Connect the GND pin of the sensor to the GND pin of Arduino.

Connect a buzzer (08) led to a digital pin for gas-level.

Open the Arduino IDE

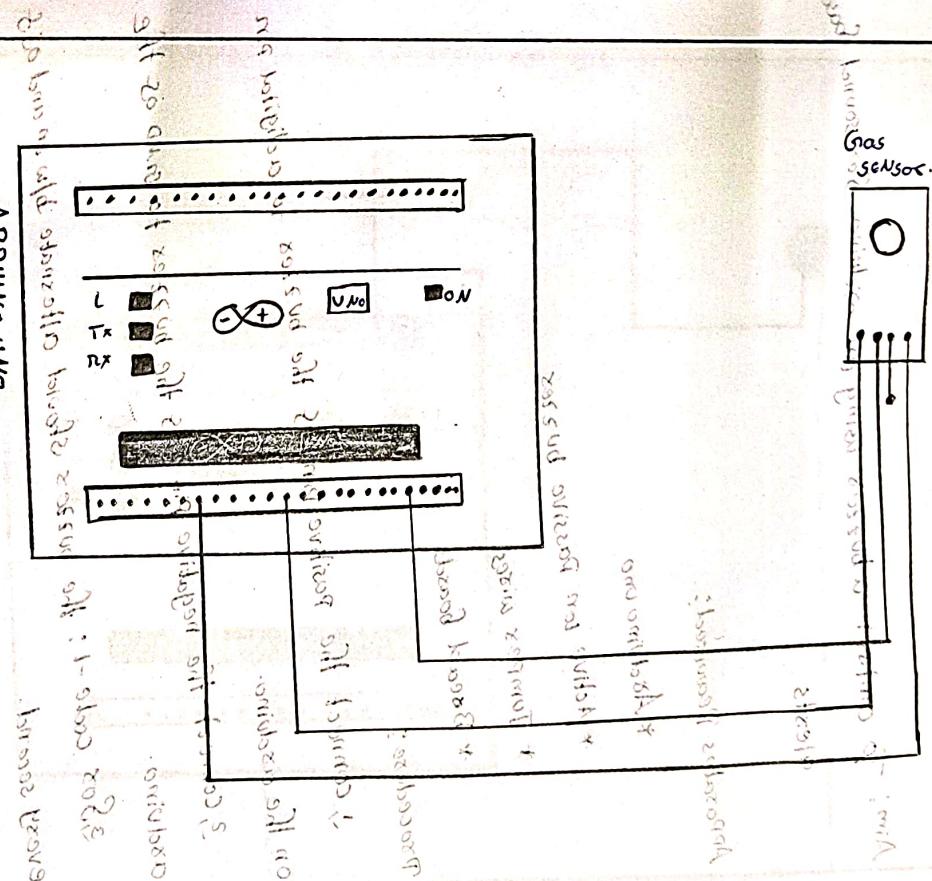
Open the Serial monitor to observe gas-level readings.

Output:
The sensor detects gas levels and outputs analog reading.

Result:
Thus the above program is executed successfully by

gas sensor detection pins as soon as ever when

Analog is triggered when gas levels exceed the set threshold.



Thus the above program is executed successfully by

3. Interfacing a buzzer with arduino uno.

Aim: To control a buzzer using an arduino for sound alerts.

Apparatus Required:

* Arduino uno

* Active or passive buzzers

* Jumper wires

* Bread boards

Procedure:

1. Connect the positive pin of the buzzer to a digital pin on the arduino.

2. Connect the negative pin of the buzzer to ground as the arduino.

3. For code-1: the buzzer should alternate blow on and off every second.

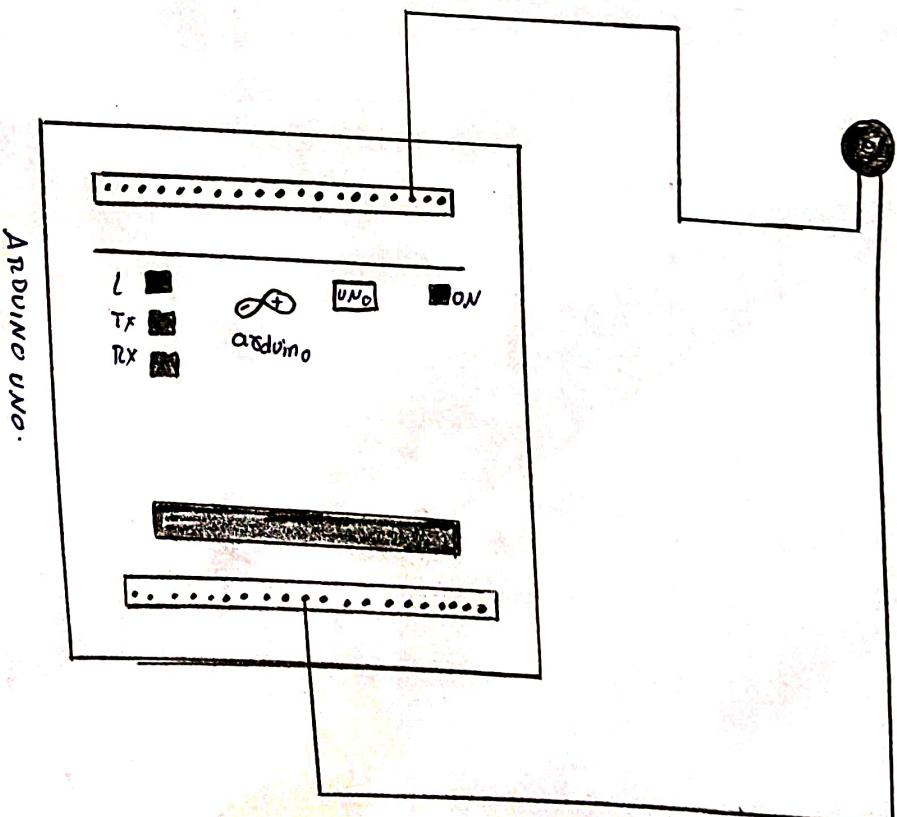
4. For code-2: the buzzer will generate a periodic tone.

5. Adjust the tone frequency for different sound patterns.

Output:

The buzzer emits a continuous (or) periodic sound.

The sound varies based on code written and the buzzer serves as an audio alert mechanism.



Result:

Thus the above program is executed successfully by interfacing a buzzer.