B.Sc.III: Semester- V Paper- DSE 1005E1 Linear Integrated Circuits, 8051 Microcontroller Interfacing and Embedded C

Section II: 8051 Microcontroller Interfacing and Embedded C

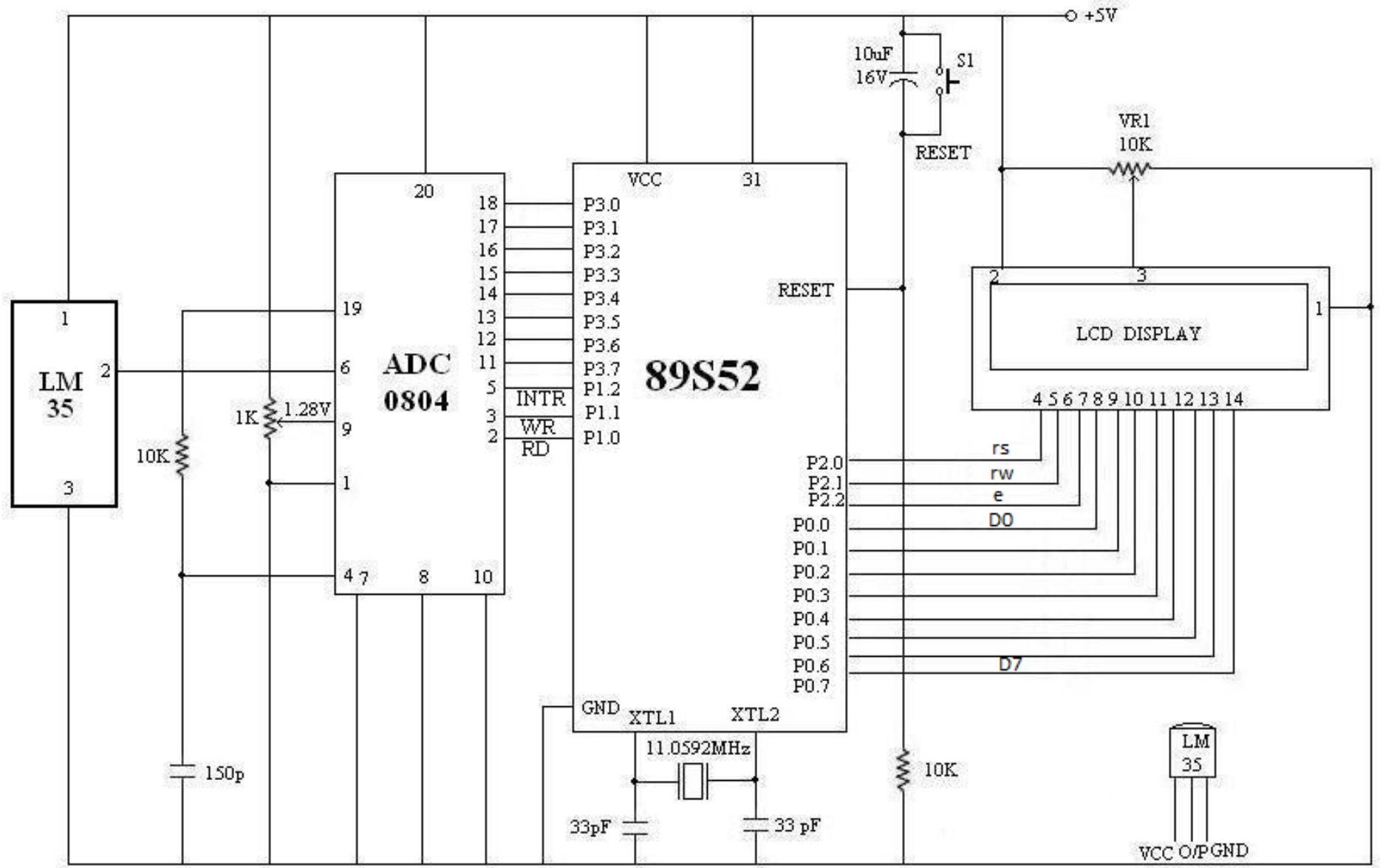
UNIT 4: Applications of 8051

Presented By:
Dr. C. B. Patil
Vivekanand College (Autonomous), Kolhapur

Unit 3: Applications of 8051:

Case study's: i) Gate Emulator (Logic Gate study using microcontroller) ii) Water level controller iii) speed control of DC motor iv) Temperature measurement using LM35, ADC0804, LCD. v)Bluetooth module interfacing. Vi) Speed control of Stepper Motor

Temperature Measurement System



- A precision IC sensor LM35 is used here for measurement of temperature. Its resolution is 10 mV/ºC. It does not required external calibration. The output of LM35 is in voltage (analog) form. It is converted into digital form by using ADC 0804.
- The microcontroller port P3 is used here for read data from ADC.
- The port pin P1.0, P1.1 and P1.2 are connected to the RD, WR and INTR pins of the ADC respectively.
- P0 is used for send command and data to the LCD.
- The port pin P2.0, P2.1 and P2.2 are connected to the RS, RW and E pins of the LCD respectively.
- The ADC resolution is adjusted to 10mV/step by using potentiometer connected at Vref/2 pin(1.28 V)

```
//---------------------- Temperature Measurement System --------------------//
#include <REGX51.H>
#define port P0       // LCD data lines are connected to Port P0
sbit rs=P2^0;         // LCD rs pin connected at P2.0
sbit rw=P2^1;         // LCD rw pin connected at P2.1
sbit en=P2^2;         // LCD en pin connected at P2.2
sbit rd=P1^0;         // ADC rd pin connected at P1.0
sbit wr=P1^1;         // ADC wr pin connected at P1.1
sbit intr=P1^2;       // ADC intr pin connected at P1.2
void ini();
void lcdmd(unsigned char);    //LCD Command function
void MSDelay(unsigned int);  // delay function
void LCDData(unsigned char);              //LCD data function
void DISPLAY(unsigned char); // Display function
void CONVERT(unsigned int); // data conversion function
```

```
void main()
{               unsigned int value,i,j;
                unsigned char msg1[]="WELCOME";
                unsigned char msg2[]="TEMPERATURE=";
                ini();
                MSDelay(1);
                lcdmd(0x84);
                MSDelay(1);
                for(i=0;i<=6;i++)
                                {               j=msg1[i];
                                                LCDData(j);
                                }
                lcdmd(0xC0);
                MSDelay(1);
                for(i=0;i<=11;i++)
                                {               j=msg2[i];
                                                LCDData(j);
                                }
                intr=1;
                rd=1;
                wr=1;
                while(1)
                {
                                wr=0;
                                wr=1;
                                while(intr==1);
                                rd=0;
                                value=P3;
                                CONVERT(value);
                                rd=1;
                                MSDelay(10);
                                MSDelay(10);
                                MSDelay(10);
                }
}
```

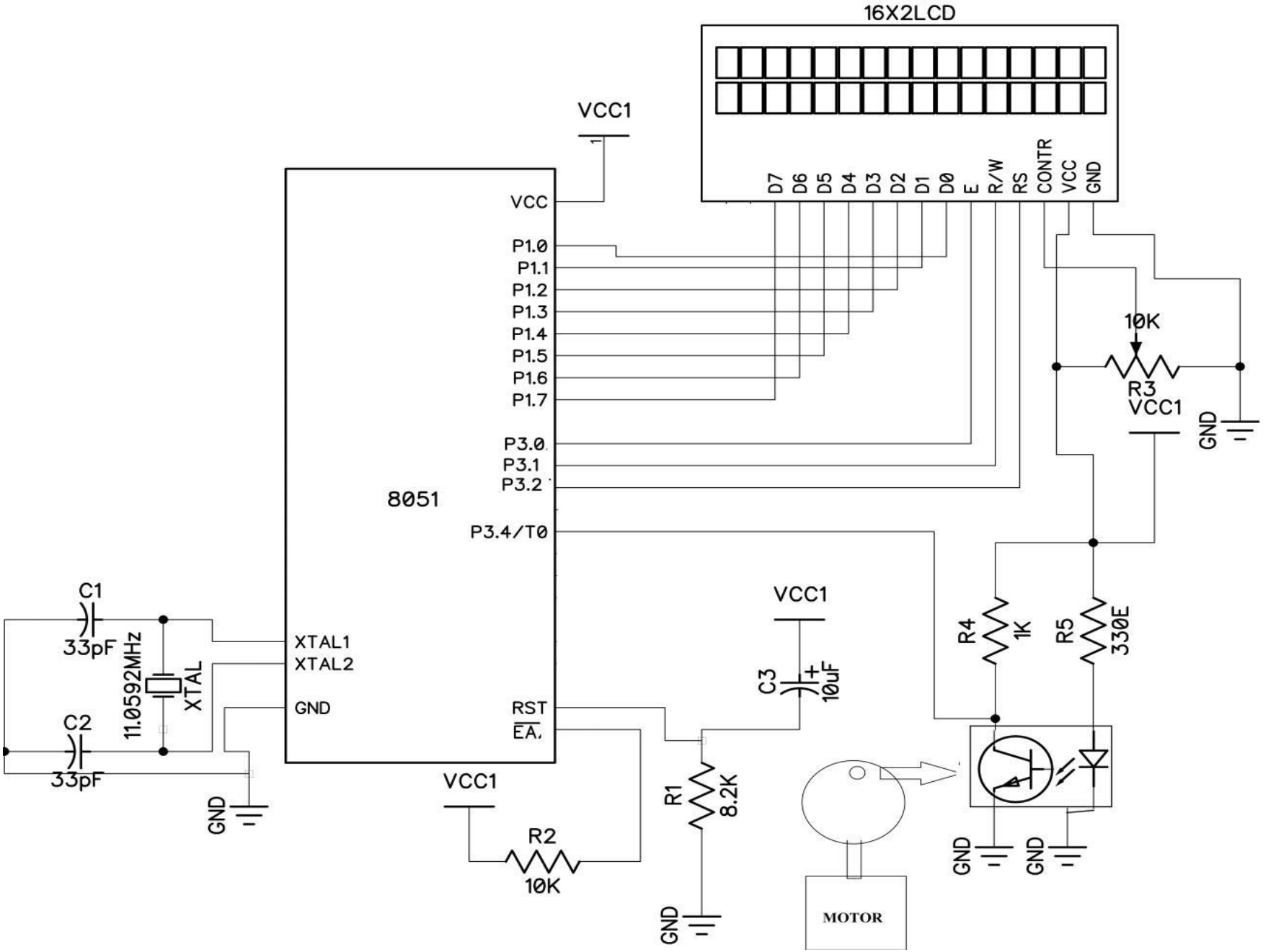
```
void CONVERT(unsigned int e)
{
        unsigned char x,d1,d2,d3, units, tens, hundreds;
        x=e/10;
        d1=e%10;
        d2=x%10;
        d3=x/10;
        units=(d1+0x30);    //ASCII conversion
        tens=(d2+0x30);     //ASCII conversion
        hundreds=(d3+0x30);          //ASCII conversion
        lcdmd(0xCC);                 // Cursor at line 2, position 13
        LCDData (hundreds);
        LCDData (tens);
        LCDData (units);
        LCDData('C');
        MSDelay(10);
}
```

```
void ini()
        {
                lcdmd(0x38);
                lcdmd(0x0C);
                lcdmd(0x01);
                lcdmd(0x06);
                lcdmd(0x80);
                LCDData(' ');
                lcdmd(0x90);
                LCDData(' ');
                lcdmd(0xC0);
                LCDData(' ');
                lcdmd(0xD0);
                LCDData(' ');
        }
```

```
void lcdmd(unsigned char value)
        {       port=value;
                rs=0;
                rw=0;
                en=1;
                en=0;
                MSDelay(1);
                return;
        }
        void MSDelay(unsigned int itime)
        {       unsigned int i,j;
                for(i=0;i<=itime;i++)
                for(j=0;j<1275;j++);
        }
         void LCDData(unsigned char value)
        {       port=value;
                rs=1;
                rw=0;
                en=1;
                en=0;
                MSDelay(1);
                return;
        }
```

Speed Measurement of the Motor:

The complete circuit schematic of the speed measurement system for the motor using 8051 is as shown in fig.4.3 . The system is developed by using opt o-coupler, microcontroller 8051 and LCD. Here the wheel (with one hole) is mounted on the shaft of the motor. It is rotated in between opto- coupler. During the rotations the opto- coupler gives pulses. These pulses are applied at input pin of the counter T0 of 8051. The microcontroller counts the external pulses in certain time delay and the measured pulses are send to the LCD. It displays the result in the RPM. The port 3 pin P3.4 is used to count pulses for counter 0 and using internal timer register the delay is generated. The data lines of LCD are connected to P1. The control lines RS, RW and E are connected to port 3 pins P3.2, P3.1 and P3.0 respectively. It is useful to control the speed of the motor in RPM (Rotation per Minute).


16X2LCD
D7
D6
D5
D4
D3
D2
D1
D0
E
R/W
RS
CONTR
VCC
GND
VCC1
8051
VCC
P1.0
P1.1
P1.2
P1.3
P1.4
P1.5
P1.6
P1.7
P3.0
P3.1
P3.2
P3.4/T0
XTAL1
XTAL2
GND
RST
EA
10K
R3
VCC1
GND
C1
33pF
11.0592MHz
XTAL
C2
33pF
GND
VCC1
C3
10uF
R4
1K
R5
330E
R1
8.2K
GND
VCC1
R2
10K
MOTOR
GND
GND

```
;MOTOR SPEED MEASURMENT SYSTEM USING 8051
;SYSTEM IS USEFUL TO MEASURE THE SPEED UPTO 255 RPM
;-------------------------------------------------------------------------------------------------
LDATA EQU P1        ;LCD data lines connected at P1
RS EQU P3.2         ;LCD RS pin connected at P3.2
RW EQU P3.1         ;LCD RW pin connected at P3.1
E EQU P3.0          ;LCD E pin connected at P3.0
BUSY EQU P1.7        ;LCD BUSY bit connected at P1.7
;-------------------------------------------------------------------------------------------------
ORG 0000H
LJMP MAIN ;SKIP SUBR. OF TIMER, SERIAL AND INTERRUPT
;-------------------------------------------------------------------------------------------------
ORG 0030H
            ACALL LCD_INI
BACK:       ACALL COUNT
            SJMP BACK
;-------------------------------------------------------------------------------------------------
```

```
COUNT:  MOV TMOD, #14H
        MOV TL0,#00H
        SETB TR0 ;START THE COUNTER T0
        MOV R0,#200 ;COUNT PULSES FOR 5 S
        HERE2: MOV R1,#5
        HERE1: MOV TH1,#0EEH
        MOV TL1,#00H
        SETB TR1
AGAIN:  JNB TF1, AGAIN
        CLR TR1
        CLR TF1
        DJNZ R1,HERE1
        DJNZ R0,HERE2
        MOV A, TL0 ; LOAD COUNTED PULSES IN ACCUMULATOR
        MOV B,#12
        MUL AB ;convert count For 60 Seconds
        ACALL RESULT_DISPLAY
        RET
```

;This program provides count pulses for 5 S this is converted to the 60 S by multiplying 12 to count.

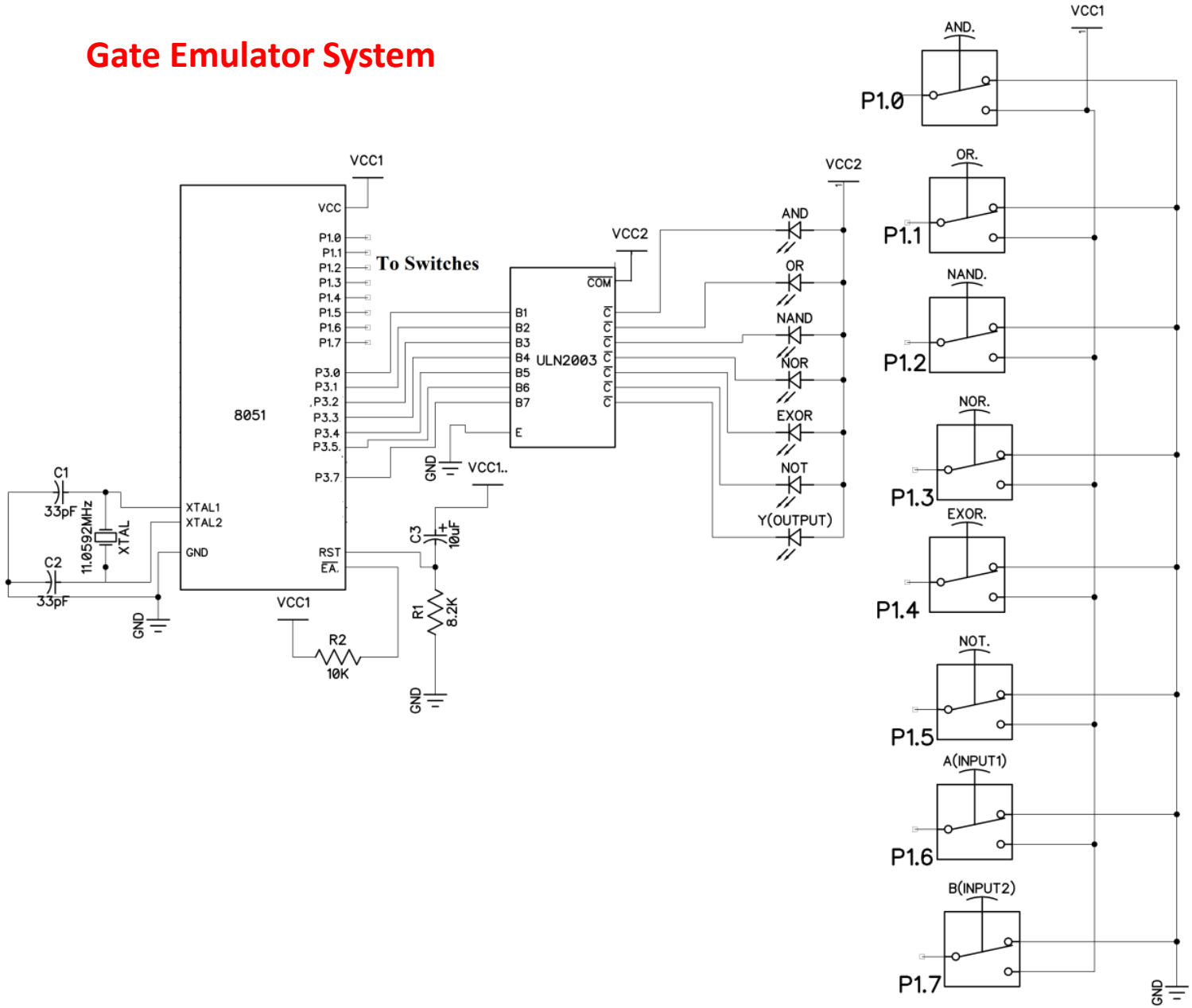
;----------

```
LCD_INI:  MOV A,#38H ; Initialize LCD 2 lines 5 x 7 matrix
          ACALL COMMAND ; Command subroutine Call
          MOV A,#0EH ;Display On cursor On
          ACALL COMMAND ; Command subroutine Call
          MOV A,#01H ;Clear LCD
          ACALL COMMAND ; Command subroutine Call
          MOV A,#06H ;Shift cursor right
          ACALL COMMAND ; Command subroutine Call
          MOV A,#84H ; Cursor at line 1, position 4
          ACALL COMMAND ; Command subroutine Call
          RET ; Return to main program
;-------------------------------------------------------------------------------------------------
COMMAND: ACALL READY ; subroutine Checking display LCD status (ready/busy)
          MOV LDATA,A ; Send command to LCD on data pin
          CLR RS ; RS=0 for command
          CLR RW ; RW=0 for LCD write operation
          SETB E ;Make enable pin E high to Low
          NOP
          NOP
          CLR E
          RET ; Return to main program
;-----------------------------------------------------
```

```
-------------------------------------------------
DISPLAY:  ACALL READY ;subroutine Checking display LCD status (ready/busy) MOV
          LDATA,A ; Send data to LCD on data pin
          SETB RS ; RS=1 for data
          CLR RW ; RW=0 for write operation
          SETB E ; Make enable pin E high to low
          NOP
          NOP
          CLR E
          RET ; Return to main program
READY: SETB BUSY ;Make busy pin as an input
          CLR RS ; RS=0 for command
          SETB RW ; RW=0 for write operation
          BACK: CLR E ; Make enable pin E low to high
          NOP
          NOP
          SETB E ; Checking LCD status
          JB BUSY,BACK
          RET ; Return to main program
;-------------------------------------------------------------
```

```
RESULT_ DISPLAY: MOV R0,A ;Accumulator store at safe place
                 MOV A,#84H ; Cursor at line 1, position 4
                 ACALL COMMAND ; Command subroutine Call
                 MOV A,R0 ;Decimal conversion of count
                 MOV B,#100
                 DIV AB
                 ORA A,#30H ; ASCII conversion
                 ACALL DISPLAY ;display subroutine call
                 MOV A,B
                 MOV B,#10
                 DIV AB
                 ORA A,#30H ; ASCII conversion
                 ACALL DISPLAY ;display subroutine call
                 MOV A,B
                 ORA A, #30 ;ASCII conversion
                 ACALL DISPLAY ;display subroutine call
                 MOV A, #‘R’
                 ACALL DISPLAY
                 MOV A, #‘P’
                 ACALL DISPLAY
                 MOV A, #‘M’
                 ACALL DISPLAY
                 RET
```

Gate Emulator System

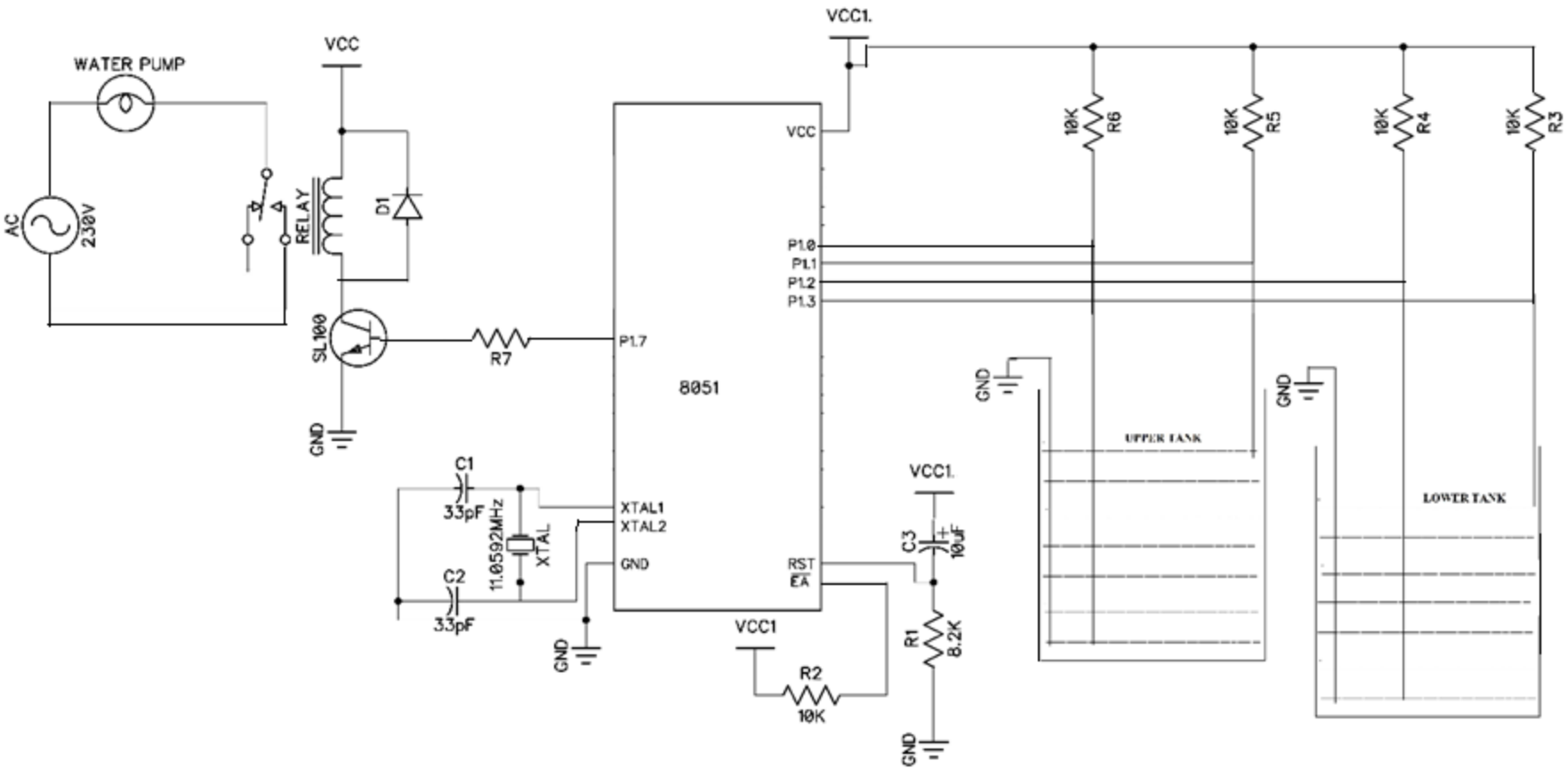


```
;-----------------------------------------------------------------------------------------------------------
;GATE EMULATER SYSTEM USING 8051
;-----------------------------------------------------------------------------------------------------------
ORG 0000H
LJMP MAIN ;SKIP SUBROUTINE VECTOR OF TIMER, SERIAL AND INTERRUPT
;-----------------------------------------------------------------------------------------------------------
ORG 0030H
MAIN: MOV P1,#0FFH ;MAKE P1 AS AN INPUT
HERE: ACALL SELECT ;SELECTED GATE INDICATION
JB P1.0,HERE1
ACALL AND_1 ;FOR LOGICAL AND OPERATION
HERE1: JB P1.1,HERE2
ACALL OR_1 ;FOR LOGICAL OR OPERATION
HERE2: JB P1.2,HERE3
ACALL NAND_1 ;FOR LOGICAL NAND OPERATION
HERE3: JB P1.3,HERE4
ACALL NOR_1 ;FOR LOGICAL NOR OPERATION
HERE4: JB P1.4,HERE5
ACALL EXOR_1 ;FOR LOGICAL EXOR OPERATION
HERE5: JB P1.5,HERE6
ACALL NOT_1 ;FOR LOGICAL NOT OPERATION
HERE6: SJMP HERE
;-----------------------------------------------------------------------------------------------------------
SELECT: MOV A,P1 ;SELECTED GATE INDICATION ON P3
CPL A
MOV P3,A ;LED CONNECTED IN SINKING MODE
RET
;-----------------------------------------------------------------------------------------------------------
```

```
AND_1: MOV C,P1.7
ANL C,P1.6
MOV P3.7,C
RET
;-------------------------------------------------------------------------------------------------
OR_1: MOV C,P1.7
ORL C,P1.6
MOV P3.7,C
RET
;-------------------------------------------------------------------------------------------------
NAND_1: MOV C,P1.7
ANL C,P1.6
CPL C ;NAND OPERATION
MOV P3.7,C
RET
;-------------------------------------------------------------------------------------------------
```

```
NOR_1: MOV C,P1.7
ORL C,P1.6
CPL C ;NOR OPERATION
MOV P3.7,C
RET
;-----------------------------------------------------------------------------------------------
EXOR_1: MOV A,P1
RL A
MOV B,P1
XRL A,B
RLC A
MOV P3.7,C
RET
;-----------------------------------------------------------------------------------------------
NOT_1: MOV C,P1.7
CPL C
MOV P3.7,C
RET
```

Water Level Controller



The complete circuit schematic of the water level controller using 8051 is as shown in fig 4.5. Here port P1 pins P1.0, P1.1 ,P1.2 and P1.3 are used to monitor the statues of the tanks. Here the system is developed by considering two tanks upper and lower.

P1.0 – Upper tank lower electrode,

P1.1- Upper tank upper electrode,

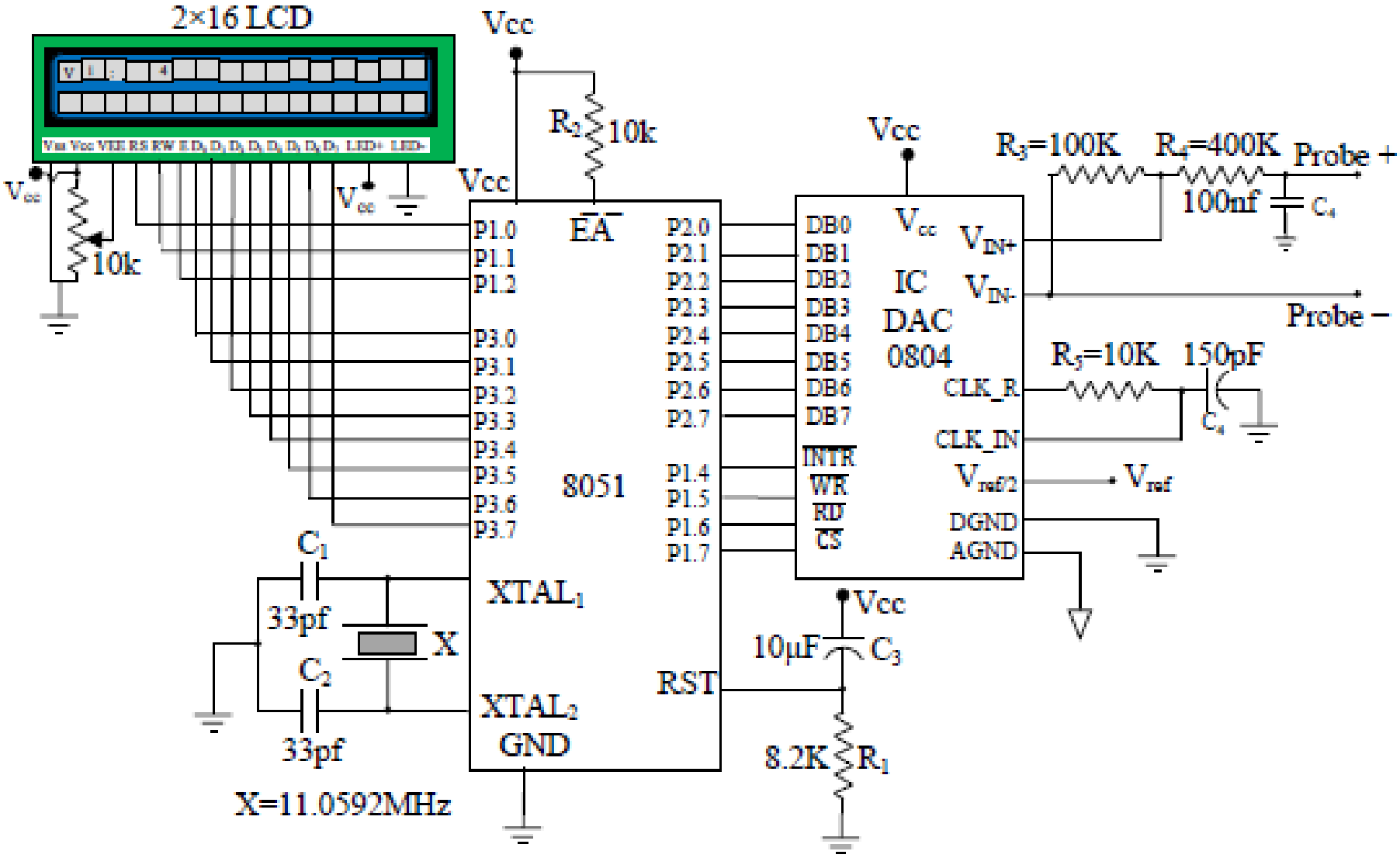
P1.2- Lower tank lower electrode,

P1.3- Lower tank upper electrode.

The electrodes in water indicates logic ‘0’ and in air indicates logic ‘1’. The water pump(motor) is connected at port P1 pin P1.7. The motor will be ON for P1.7= ‘1’ and OFF for P1.7= ‘0’. The motor is used to pumping water from lower tank to upper tank. The Microcontroller 8051 check the statues of the upper tank and lower tank according to maintain the level of water the upper tank.

```
;-----------------------------------------------------------------------------------------------------
;WATER LEVEL CONTROLLER USING 8051
;-----------------------------------------------------------------------------------------------------
ORG 0000H
LJMP MAIN            ;SKIP SUBROUTINE VECTOR OF TIMER, SERIAL AND INTERRUPT
;-----------------------------------------------------------------------------------------------------
ORG 0030H
MAIN: MOV P1,#0FH            ;MAKE P1 PIN P1.0- P1.3 AS AN INPUT
MOTOR_OFF: CLR P1.7          ; MAKE MOTOR OFF
JB P1.0 MOTOR_ON             ; IF UPPER TANK EMPTY MAKE MOTOR ON
SJMP MOTOR_OFF               ; OTHERWISE MAKE MOTOR OFF
;-----------------------------------------------------------------------------------------------------
MOTOR_ON: SETB P1.7          ;MAKE MOTOR ON
BACK: JB P1.2, MOTOR_OFF     ;IF LOWER TANK EMPTY MAKE MOTOR OFF
JNB P1.1, MOTOR_OFF          ;IF UPPER TANK FULL MAKE MOTOR OFF
SJMP BACK                    ; CONTINUES
```

Digital Voltmeter



Assume that, the 4V is applied to input of probe (V_{ip}) then voltage divider circuit output (Y) is

$$Y = \frac{R_3}{R_3 + R_4} \times V_{ip} = \frac{100K}{100K + 400K} = \frac{100K}{500K} = \frac{1}{5} \times 4 = 0.8V$$

Here the output of voltage divider circuit is 0.8V, which is applied to 10 bit ADC IC 0804. Total steps of this ADC are $2^8 = 256$ (0-255) while each step size is 5V/255 = 19.6mV. The equation for the digital output of ADC0804 is

$$\therefore D_{Out} = \frac{V_{In}\,(Y)}{\text{Step size}} = \frac{0.8}{19.6mV} = 40.8163 \quad (\text{in binary} - 0010\ 1000\ )$$

ADC provides count 40 to microcontroller, and microcontroller divides the received D_{out} count of by 10. Therefore 40/10 = 4 which is used for displaying on digital display unit. From above example, it is seen that if 4V is applied to the input probes then 0.8V gets at the output of voltage divider circuit. ADC takes this 0.8V and provides count 0010 1000

```
START:  LDATA EQU P3          ; LCD data lines are connected to P3
        RS EQU P1.0           ; RS pin of LCD is connected to P1.0
        RW EQU P1.1           ; RW pin of LCD is connected to P1.1
        E EQU P1.2            ; E pin of LCD is connected to P1.2
        BUSY EQU P3.7         ; BUSY pin of LCD is connected to  P3.7
        ADATA EQU P2          ; ADC data lines are connected to P2
        ADC_CS EQU P1.7       ; CS pin of ADC is connected to P1.7
        ADC_RD EQU P1.6       ; RD pin of ADC is connected to P1.6
        ADC_WR EQU P1.5       ; WR pin of ADC is connected to P1.5
        ADC_INTR EQU P1.4     ; INTR pin of ADC is connected to P1.4
-------------------------------------------- ------------------------------------------
ORG 0000H
        LJMP MAIN ; Skip subroutine of timer serial and interrupt
-------------------------------------------- ------------------------------------------
```

```
ORG 0030H
MAIN:   MOV ADATA, #0FFH ; Making ADC port (P2) as an input
        ACALL LCD_INI
BACK:   CLR ADC_CS       ; ADC CS = 0 (Select Chip)
        SETB ADC_RD      ; ADC RD = 1
        CLR ADC_WR       ; Apply Low to High pulse to ADC
        SETB ADC_WR      ;WR pin for start conversion
WAIT:   JB ADC_INTR, WAIT ; If ADC conversion complete INTR=1
        CLR ADC_CS
        CLR ADC_RD       ; Converted digital data gets at data lines
        MOV A, ADATA     ; Read conversion data from ADC
        MOV B,#10        ; ADC data calibration
        DIV AB
        ACALL RESULT_DISPLAY
        SJMP BACK
---------- ----------
```

```
LCD_INI:  MOV A, #38H        ; Initialize LCD 2 lines 5x7 matrix
          ACALL COMAND       ; Command subroutine call
          ACALL DELAY        ; Give LCD some time
          MOV A, #0FH        ; Display ON cursor Blinking
          ACALL COMAND       ; Command subroutine call
          ACALL DELAY         ; Give LCD some time
          MOV A, #01H        ; Clear LCD
          ACALL COMAND       ; Command subroutine call
          ACALL DELAY        ; Give LCD some time
          MOV A, #06H        ; Shift cursor right
          ACALL COMAND       ; Command subroutine call
          ACALL DELAY         ; Give LCD some time
          MOV A, #84H        ; Cursor at line1, Position 4
          ACALL COMAND       ; Command subroutine call
          ACALL DELAY        ; Give LCD some time
          RET                 ; Return to main program
------------------- ------------------------------------------------------------------------------
```

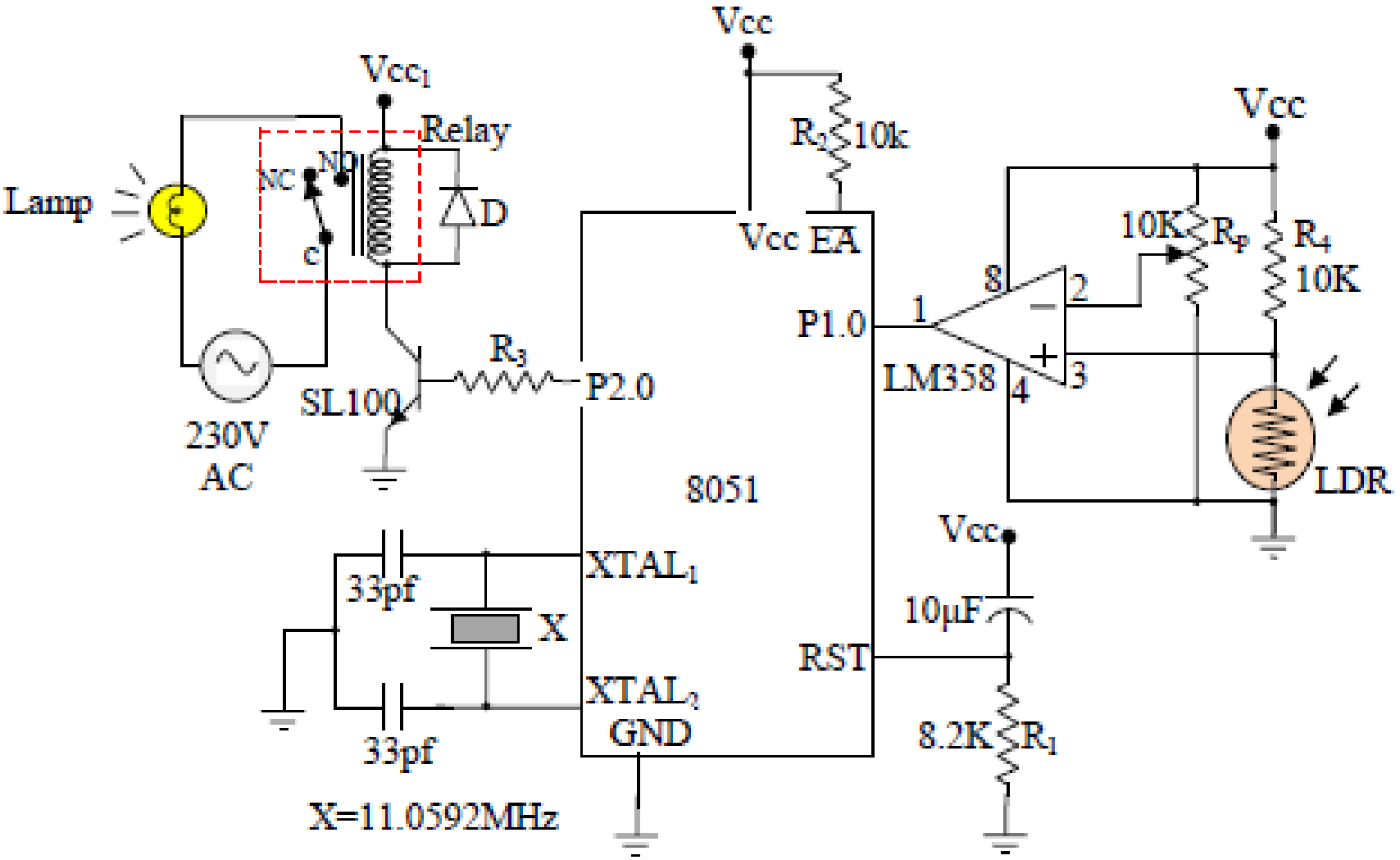
```
COMMAND: ACALL READY     ; Subroutine checking LCD status (ready or busy)
         MOV LDATA, A    ; Send command to LCD on data pin
         CLR RS          ; RS=0 for command
         CLR RW          ; RW=0 for LCD write operation
         SETB E          ; Make Enable pin E high to low
         NOP
         NOP
         CLR E
         RET             ; Return to main program
---------------------------------------- ----------------------------------------
DELAY:   MOV R3, #50     ; 50 or higher for fast CPUs
HERE2:   MOV R4,#255     ;R4 = 255
HERE:    DJNZ R4,HERE    ;Stay until R4 becomes 0
         DJNZ R3,HERE2
         RET
---------------------------------------- ----------------------------------------
```

```
DISPLAY:  ACALL READY       ; Subroutine checking LCD status (ready or busy)
          MOV LDATA, A      ; Send command to LCD on data pins
          SETB RS           ; RS = 1 for data
          CLR RW            ; RW = 0 for LCD write operation
          SETB E            ; Make enable pin E high to low
          NOP
          NOP
          CLR E
          RET               ; Return to main program
---------------------------------------------- ----------------------------------------------
READY:    SETB BUSY          ; Make busy pin as an input
          CLR RS            ; RS=0 for command
          SETB RW           ; RW=0 for write operation
          CLR E             ; Make Enable pin E low to high
          NOP               ; No Operation
          NOP               ; No Operation
          SETB E
BACK1:    JB BUSY, BACK1    ; Checking LCD status
          RET               ; Return to main program
---------------------------------------------- ----------------------------------------------
```

RESULT_DISPLAY:

```
        MOV R0, A              ; Accumulator store at safe place
        MOV A, #84H            ; Cursor at line1, Position 4
        ACALL COMAND           ; Command subroutine call
        ACALL DELAY            ; Give LCD some time
        MOV A, R0              ; Decimal conversion of count
        MOV B, #10
        DIV AB
        ORA A #30H             ; ASCII conversion
        ACALL DISPLAY          ; Display subroutine call
        ACALL DELAY            ; Give LCD some time
        MOV A, B
        ORA A, #30H            ; ASCII conversion
        ACALL DISPLAY          ; Display subroutine call
        ACALL DELAY            ; Give LCD some time
        MOV A, #80H            ; Cursor at line1, Position 4
        ACALL COMAND           ; Command subroutine call
        ACALL DELAY            ; Give LCD some time
        MOV A, # 'V'           ; Send letter 'V' for display
        ACALL DISPLAY          ; Display subroutine call
        ACALL DELAY            ; Give LCD some time
        MOV A, # 'i'           ; Send letter 'i' for display
        ACALL DISPLAY          ; Display subroutine call
        ACALL DELAY            ; Give LCD some time
        MOV A, # ':'           ; Send letter ':' for display
        ACALL DISPLAY          ; Display subroutine call
        ACALL DELAY            ; Give LCD some time
        RET
```

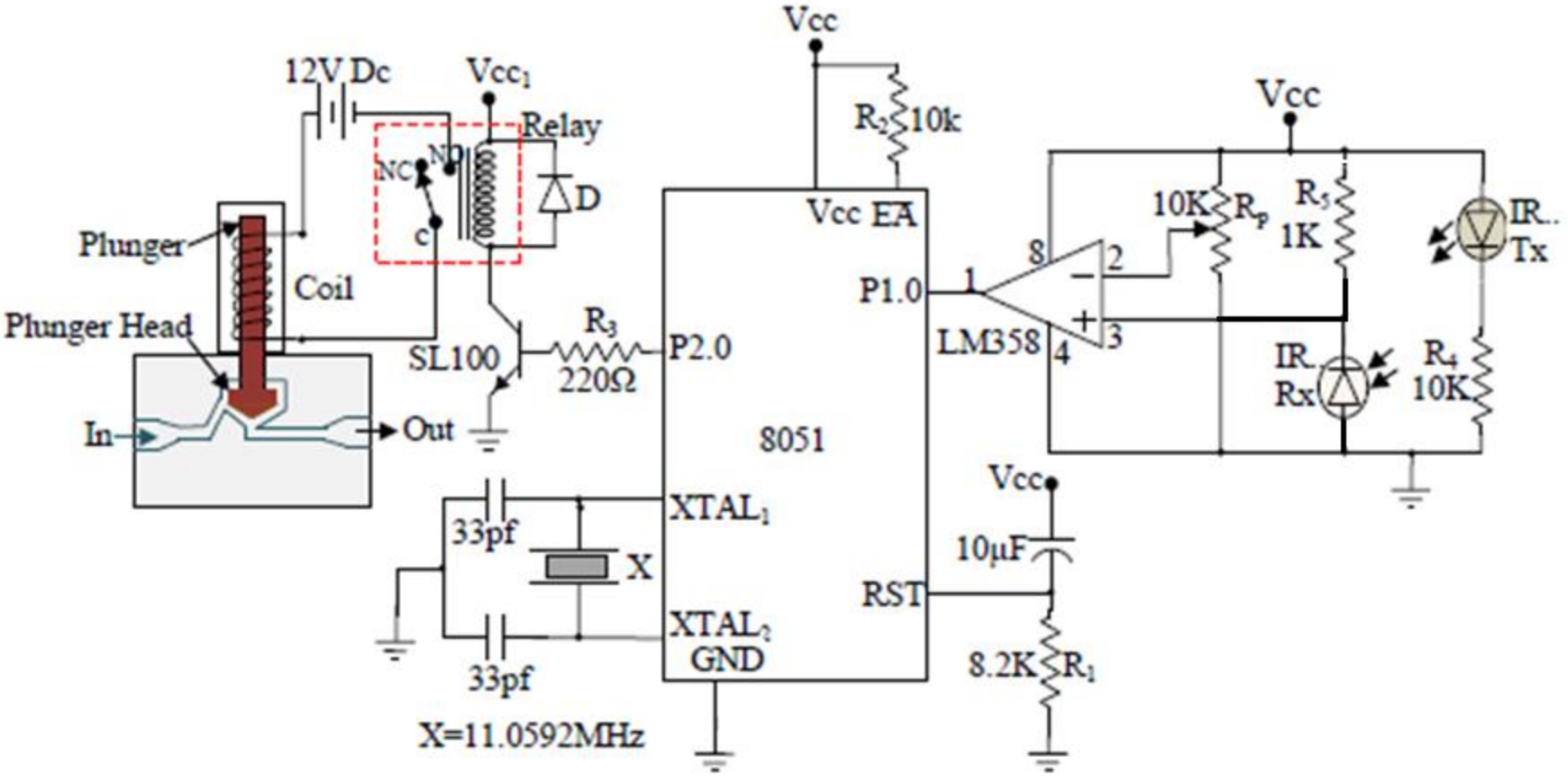
Automatic Street light control System



Automatic Street light control System

```
        SETB P1.7          ;Make P1.7 as input
AGAIN:  JB P1.7 HERE       ;Jump if P1.7=1
        CLR P2.0           ;Lamp is OFF
        SJMP AGAIN         ;Keep monitoring
HERE:   SETB P2.0          ;Lamp is ON
        SJMP AGAIN         ;Keep monitoring
```

Automatic Basin Control (using IR and solenoid switch)



Automatic Basin Control (using IR and solenoid switch)

```
        SETB P1.0          ;Make P1.0 as input
AGAIN:  JB P1.0 HERE       ;Jump if P1.0=1
        CLR P2.0           ;Solenoid Valve is OFF
        SJMP AGAIN         ;Keep monitoring
HERE:   SETB P2.0          ; Solenoid Valve is ON
        SJMP AGAIN         ;Keep monitoring
```