

PC-BASED 7-SEGMENT ROLLING DISPLAY

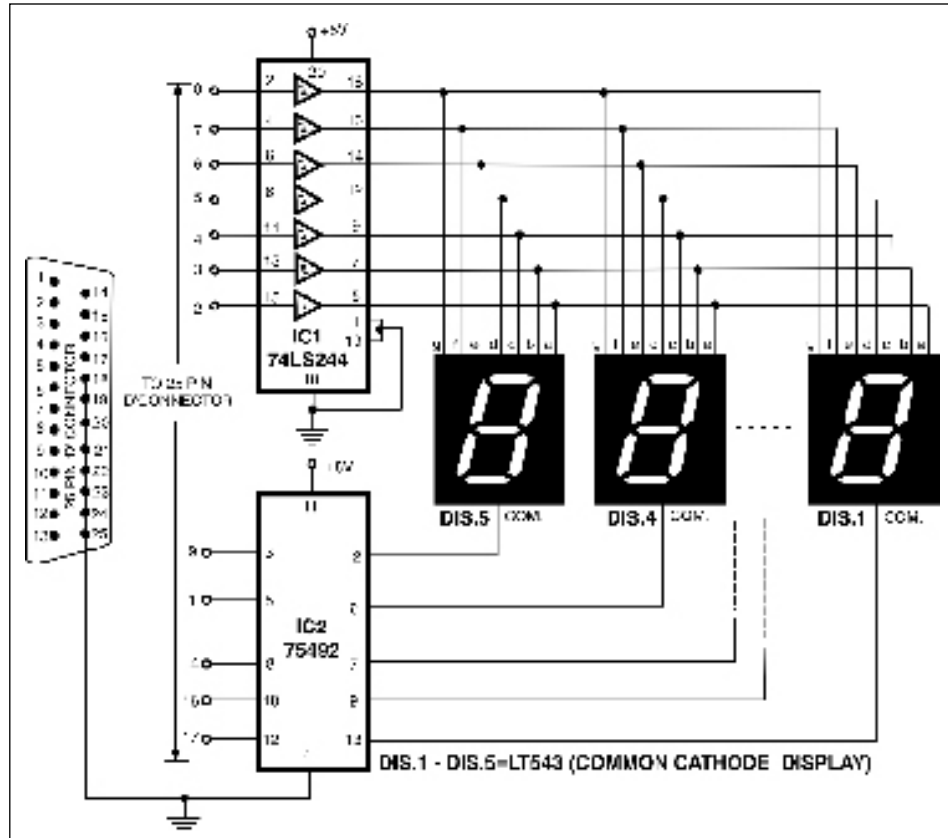
It is very interesting and convenient to be able to control everything while sitting at your PC terminal. Here, a simple hardware circuit and software is used to interface a 7-segment based rolling display.

The printer port of a PC provides a set of points with some acting as input lines and some others as output lines. Some lines are open collector type which can be used as input lines. The circuit given here can be used for interfacing with any type of PC's printer port.

The 25-pin parallel port connector at the back of a PC is a combination of three ports. The address varies from 378H-37AH. The 7 lines of port 378H (pins 2 through 8) are used in this circuit to output the code for segment display through IC1. The remaining one line of port 378H (pin 9) and four lines of port 37AH (pins 1, 14, 16, 17) are used to enable the display digits (one at a time) through IC2.

The bits D0, D1 and D3 of port 37AH connected to pins 1, 14 and 17 of 'D' connector are inverted by the computer before application to the pins while data bit D2 is not inverted. Therefore to get a logic high at any of former three pins, we must send logic 0 output to the corresponding pin of port 37AH.

Another important concept illustrated by the project is the time division multiplexing. Note that all the five 7-segment displays share a common data bus. The



PC places the 7-segment code for the first digit/character on the data bus and enables only the first 7-segment display. After delay of a few milliseconds, the 7-segment code for the digit/character is replaced by that of the next character/digit, but this time only second display digit is enabled.

After the display of all characters/digits in this way, the cycle repeats itself

over and over again. Because of this repetition at a fairly high rate, there is an illusion that all the digits/characters are continuously being displayed. DISP1 is to be physically placed as the least significant digit.

IC1 (74LS244) is an octal buffer which is primarily used to increase the driving capability. It has two groups of four buff-

PROGRAM

```

/*DISPC*** PC BASED ROLLING
DISPLAY */
/* P.R.DESHMUKH*/
#include<stdio.h>
#include<conio.h>
#include<dos.h>
#define PORTA 0x378
#define PORTB 0x37a
void main()
{
    int dno[6]={0x0a,0x09,0x0f,0x03,0x80};
    /* code for "hallo"*/
    int m[5]={0x76,0x77,0x38,0x38,0x3f};
    /*code for the selection of display*/
    int f,j;

    clrscr();
    for(f=200;f<=500;f+=100)
    {
        sound(f);
        delay(100);
    }
    nosound();
    while (!kbhit())
    {
        for (j=0;j<=4;j++)
        {
            outputb(PORTA,m[j]);
            if(j<=3)
            {
                outputb(PORTB,dno[j]);
                delay(300);
            }
            else
            {
                outputb(PORTB,0x0b);
                outputb(PORTA,m[j]);
                outputb(PORTA,(m[j] || (0x80)));
                delay(300);
            }
        }
    }
}
    
```

ers with non-inverted tri-state outputs. The buffer is controlled by two active low enable lines. IC2 (75492) can drive a maximum of six 7-segment displays. (For driving up to seven common-cathode displays one may use ULN2003 described in the previous circuit idea.)

The program for rolling display is given in the listing DISP.C above.

Whatever the message/characters to be displayed (here five characters have been displayed), these are separated and stored in an array. Then these are decoded.

Decoding software is very simple. Just replace the desired character with the binary equivalent of the display code. The display code is a byte that

has the appropriate bits turned on. For example, to display character 'L', the segments to be turned on are f, e and d. This is equivalent to 111000 binary or 38 hex.

Please note that only limited characters can be formed using 7-segment display. Characters such as M, N and K cannot be formed properly.
