# PC printer port controls frequency divider

**Bogdan Manolescu, Microelectronica, Bucharest, Romania**

BBS

In **Fig 1**'s circuit, two cascaded synchronous presetable binary counters, $IC_1$ and $IC_2$, can derive signals having a frequency of $f_{CLK}/N$. In the circuit, a simple oscillator generates $f_{CLK}$; however, you can substitute any triggerable source.

An IBM PC supplies the binary-coded integer divisor N (N=255 max) via eight pins of its printer port. Two additional control lines (pins 1 and 14 of the printer port) provide start and reset functions. The signal that starts the oscillator (COM=0) also enables the first counter, $IC_1$.

The counters, wired to count down, activate the overflow output of $IC_2$ when the counters reach zero. The overflow signal then enables the counters' parallel loading of the integer divisor N. The Turbo C++ program in **Listing 1** controls the frequency dividers' operation.
(DI #1431)

EDN

**To Vote For This Design, Circle No. 346**

**Listing 1—Frequency-divider control program**

```
#include <stdio.h>
#include <conio.h>
#include <dos.h>
#include <ctype.h>

#define     OUT_PORT  0x378  /* printer output  port address */
#define     CTRL_PORT 0x37A  /* printer control port address */

int main(void)
{
  int n; char c;
  for (;;)  {
    clrscr();
    printf("Input the divisor (between 1 and 255): ");
    scanf("%d", &n);
    printf("\nStart ? (y/n)");
    if(tolower(getch()=='y')) {
      outportb(OUT_PORT,n);       //send out divisor
      outportb(CTRL_PORT,0x02);  //start oscillator and enable counter1
      printf("\n\n\tStop with any key...\
        \n\tESC to exit...");
      c=getch();
      if(c==0x1b) break;
      outportb(CTRL_PORT,0x01);  //reset the counters
      delay(1);
      outportb(CTRL_PORT,0x03);
      delay(1);
    }
  }
  return 0;
}
```

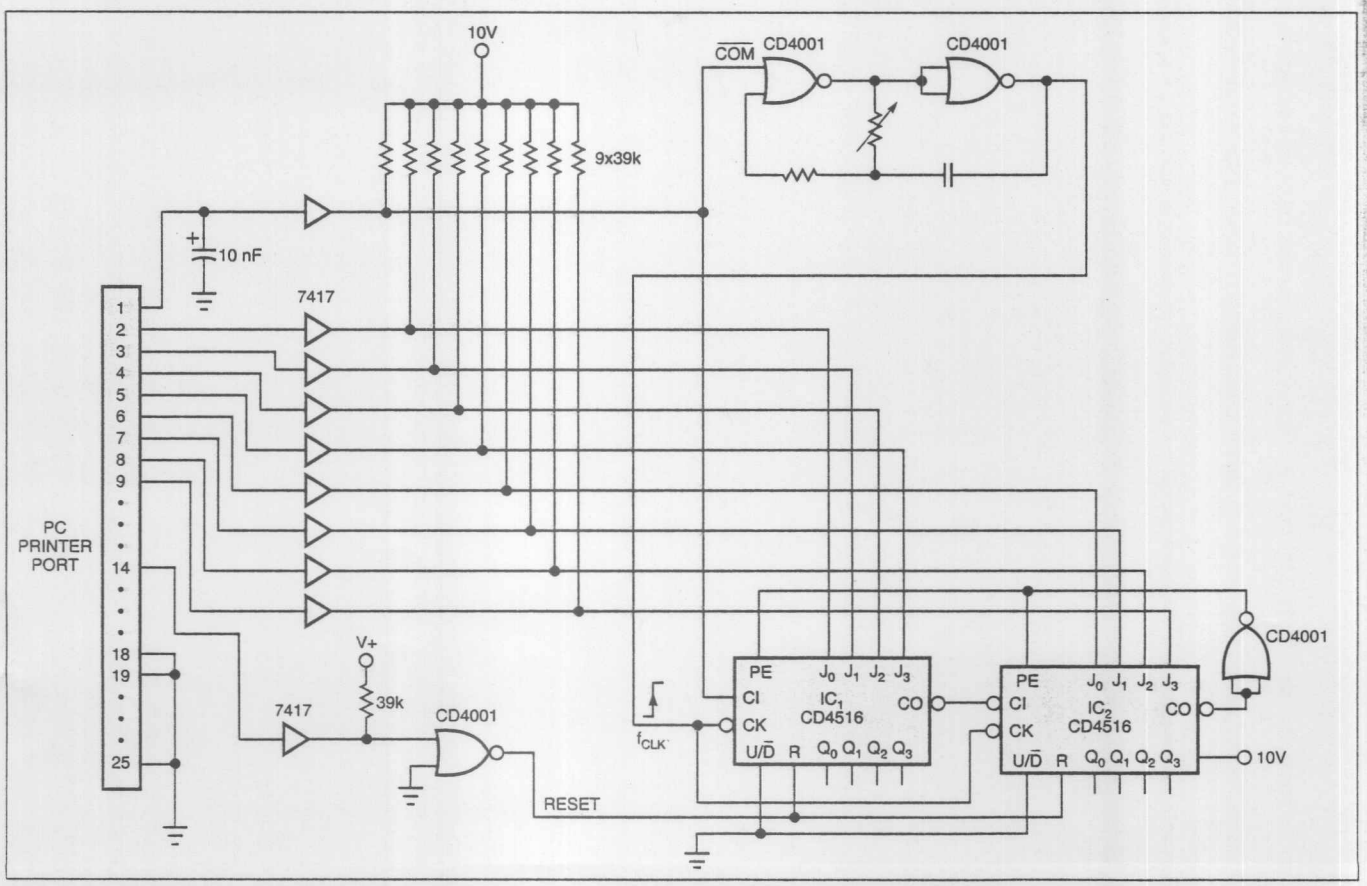


**Fig 1—An IBM PC supplies a binary-coded integer divisor circuit to this circuit's pair of synchronous, presettable binary counters.**