Circuit forms low-frequency circulator

Richard Kurzrok, Queens Village, NY

The electronic circulator made its debut ten years ago (Reference 1). It functioned at VHF as a three-port unit using a Comlinear (now part of National Semiconductor, www.nsc.com) CLC 406 operational amplifier. The circuit in Figure 1 extends the circulator’s performance to four-port operation at low frequencies, using the readily available 941 (equivalent to the ubiquitous 741) and LM318 op amps. Table 1 shows the measured data for the 741-equivalent op amp. Table 2 shows the measured data for the LM318 op amp. The four-port circulators in Figure 1 use 50 Ω impedance levels. The circuit can readily accommodate other impedance levels, such as 75 and 600 Ω. You can see that for typical circulator operation at frequencies below 50 kHz, you can use the 741-equivalent op amp. For typical operation at speeds as high as 1 MHz, you would use the LM318 op amp. The resistors in Figure 1 are metal-film units with ±1% tolerances. The circulator breadboards use open (not shielded) construction, and the components are soldered to the vector board. The ICs use commercially available sockets soldered to the vector board.

Table 1—Measured Data for 741-Equivalent Op Amp

<table>
<thead>
<tr>
<th>Frequency (kHz)</th>
<th>Forward loss (dB)</th>
<th>Reverse isolation (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.5</td>
<td>50.2</td>
</tr>
<tr>
<td>20</td>
<td>0.5</td>
<td>44.1</td>
</tr>
<tr>
<td>50</td>
<td>1.0</td>
<td>38.0</td>
</tr>
<tr>
<td>100</td>
<td>4.1</td>
<td>33.0</td>
</tr>
</tbody>
</table>

Table 2—Measured Data for LM318 Op Amp

<table>
<thead>
<tr>
<th>Frequency</th>
<th>Forward loss (dB)</th>
<th>Reverse isolation (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 Hz to 100 kHz</td>
<td>0.0</td>
<td>Greater than 56</td>
</tr>
<tr>
<td>100 kHz</td>
<td>0.1</td>
<td>Greater than 56</td>
</tr>
<tr>
<td>500 kHz</td>
<td>0.5</td>
<td>45.0</td>
</tr>
<tr>
<td>1 MHz</td>
<td>0.9</td>
<td>34.0</td>
</tr>
<tr>
<td>1.6 MHz</td>
<td>3.0</td>
<td>29.5</td>
</tr>
<tr>
<td>3.3 MHz</td>
<td>3.0</td>
<td>25.5</td>
</tr>
</tbody>
</table>
You can use the electronic four-port circulators in various applications with the fourth port terminated. You can configure baseband-amplitude and group-delay equalizers using the electronic circulator (references 2 and 3). You can also use the circuit as a low-frequency return-loss bridge or as an electronic isolator. Low-frequency op amps are available as quads with four independent op amps. You can configure a miniaturized, low-cost version of the circulator using surface-mount pc-board techniques.

**REFERENCES**

Is this the best Design Idea in this issue? Vote at [www.ednmag.com](http://www.ednmag.com).

### Use printer port as programmable frequency generator

*K Kanniappan, IGCAR, Tamil Nadu, India*

**A simple and inexpensive circuit (Figure 1) and a simple C program (Listing 1) are all you need to turn your PC’s printer port into a programmable frequency generator. Using a few low-cost and readily available components, the circuit occupies little space and is easily attachable to the printer port. The circuit has advantages over a 555-based astable multivibrator in that it eliminates the tedious task of adjusting a potentiometer while you watch a frequency counter or oscilloscope. With the circuit in Figure 1, you need only enter the desired frequency, and the PC does the rest. The circuit uses a MAX5130 low-power, programmable, 13-bit DAC, IC1; an OP07 buffer; and an AD537 VFC (voltage-to-frequency converter). The PC controls the DAC using a three-wire serial interface. It also uses the data lines D0 to D2 of the data port (0x378h) of the printer interface to send the CS (chip-select), data, and CLK (clock) signals to the DAC. Depending on the data it receives from the PC, the DAC produces a voltage output of 0 to 4.0955V in 8192 steps with a step resolu-

**Figure 1**

Turn your PC’s printer port into a programmable frequency generator with this simple circuit.

---

**Turn your PC’s printer port into a programmable frequency generator with this simple circuit.**
tion of 0.5 mV. Thus, a data word of 0×0000 produces a DAC output of 0V, and a data word of 0×1ffh produces a DAC output of 4.0955V. Using the 2.5V internal reference, the DAC output and the data input follow the equation

\[ V_{\text{OUT}} = \frac{2.5}{(\text{DATA} / 8192)} \times \text{GAIN}. \]

After IC 2 buffers it, the DAC output drives the VFC and sets its frequency output according to the following equation:

\[ f_{\text{OUT}} = \frac{V_{\text{OUT}}}{10(R_1 + R_2)C_1} \text{ Hz}. \]

For the values of \( R_1, R_2, \) and \( C \) in Figure 1 and the cited DAC-output range, the output of the VFC and hence the frequency of the programmable-frequency generator varies from 0 to 10 kHz in 8192 steps with a frequency resolution of 1.22 Hz. You choose and trim the values of \( R_1 \) and \( R_2 \) to produce a current range of 0 to 1 mA for a DAC output of 0 to 4.0955V. These values ensure good linearity (typically 0.01%) between \( V_{\text{OUT}} \) and \( f_{\text{OUT}} \). Potentiometers \( P_1 \) and \( P_2 \) adjust \( f_{\text{OUT}} \) at the lower and higher ends of the frequency range, respectively. The C program in Listing 1 obtains the desired frequency from the user and calculates the required output from the DAC to apply to the VFC. It then works out the ACTUALDATA to send to the DAC for mode control. The d2b routine converts the ACTUALDATA into 16-bit binary data. The program enables the DAC (CS) low and then serially clocks the binary equivalent of ACTUALDATA, starting one bit at a time from the MSB to the LSB, to data pin of the DAC. With the LSB set at the data pin, the low-to-high transition of the clock latches the ACTUALDATA completely into the DAC and sets \( f_{\text{OUT}} \) to the desired value. You can download Listing 1 from the Web version of this article at EDN’s Web site, www.ednmag.com. You can easily change the frequency range by changing the value of \( C \). For example, with \( R_1 \) and \( R_2 \) unchanged, you can extend the frequency to 100 kHz by changing \( C \) to 0.001 \( \mu \text{F} \) instead of 0.01 \( \mu \text{F} \). You can also increase the frequency range by using a VFC with a higher frequency capability.


---

**Listing 1—Programmable Frequency Generator**

```c
#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
#include<math.h>
#include<i808.h>
#include<dio.h>

#define CLK1 0x04 /* Clock Pulse High*/
#define CLK0 0x05 /* Clock Pulse Low*/
#define CS1 0x01 /* Chip Select high to deactivate DAC*/
#define CS0 0x06 /* Chip Select low to activate DAC*/
#define DAT1 0x02 /* Data Pulse High*/
#define DAT0 0x00 /* Data Pulse low*/

int c[16],dport,ACTUALDATA,out,k;/*Global Declarations*/
float VOUT,"DAC OUTPUT";

void d2b(unsigned int x, int c[]);/*Routine for Decimal to Binary Conversion*/
{
 int i;
 for(i=0;i<15;i++)
 c[i]=x%8;
 x=x/8;
}

float v(float x)/*Hertz to DAC output Conversion Routine*/{
 int HERTZ;
 prind("Enter the frequency within 0 to 10000Hz: ");
 scanf("%d",&HERTZ);
 VOUT=0.000040955*HERTZ;
 prind("The VOUT=",VOUT);
 return VOUT;
}

void CLOCK_DAC(void)/*Routine for clocking the DAC*/
{c=CLK1;
 outport(dport,out);"Setting the clock high"*/
delay(1);
 out=CLK0;
 outport(dport,out);"Setting the clock low"*/
delay(1);
}

void LOAD_DACDATA(int c[])/"Routine for loading actual data into the DAC"*/
{out=CS1;
 outport(dport,out);"Chip Select high to disable DAC"*/
delay(1);
 out=CS0;
 outport(dport,out);"Chip Select low to enable DAC"*/
delay(1);
 print("Data loaded into the DAC=*");
 for(k=15;k=0,k--) {
 out=CS0;
 outport(dport,out);
 print("%d",c[k]);
 delay(1);
 CLOCK_DAC();
 }
 out=CS1;
 outport(dport,doutport);d2b(0);delay(1);
}

main()
{
 int x,y;
 float y;
 unsigned int x;
 double fraction, integer, number;
 char c[];
 printf("Use Your Printer Port as a Programmable Frequency Generator");
 printf("Enter the string: ");
 printf("Enter the LPT NOT AVAILABLE! EXITING.........");
 exit(1);
 printf("Enter the Address of the port found =0x%X",dport);
 fSETS=y(VOUT&8192);/2.5,1.6384, y=y*1;
 number=y;
 fraction = mod(fraction, &integer);
 if(fraction<0.44)
incre=0;
 else incre=1;
 ACTUALDATA=16384+variable; /*Actual data including the Control Word for DAC*/
d2b(ACTUALDATA,c);
 LOAD_DACDATA(c);
 return 0;
}
```

---

Some years ago, one of the fundamental electronic instruments was the laboratory curve tracer. A CRT display would sweep out terminal behavior (current versus voltage) from which you could derive mathematical models. Classic presentations of diodes, transistors, and other devices enlightened designers about linear and nonlinear operation. From the displays, you could determine the bias points for optimum design performance. Today, however, you rarely find the classic curve tracers in the lab. Instead, you find design-simulation software, such as Spice, that’s removed from hands-on, empirical analysis. Spice models now exist for almost all electronic components. Characterization analyzers still make the voltage-current measurements but not at the design-engineer level. Rather, departments are dedicated to characterizing processes and components and incorporating these characteristics into the simulated models. The low-cost circuit in Figure 1 allows you to return to the hands-on approach by using your PC as a limited curve tracer.

The curve tracer sweeps out seven logarithmic-scaled currents from 1 \( \mu \)A to 1 mA while measuring the voltage, 0 to 5V (3.3V on some PCs), at each step. The circuit uses a programmable current source to force increasing discrete current values and samples the voltage at the IOUT terminal at each step. A classic curve tracer continuously sweeps a voltage while measuring the sourced current. The program control resides in Excel (running in Office 2000) macros that perform I/O operations through the LPT1 port of the PC. You can download the Excel program from EDN’s Web site, www.ednmag.com.

Remember the classic Tek curve tracers? You can easily configure something similar on your PC.
The program uses the free file “Input32.dll” to bit-wise control the parallel port’s digital I/O. The author of the .dll file is Jonathan Titus, editorial director of Test and Measurement World. You load CurveTracer.xls with its macros, connect the circuit of Figure 1 to the parallel port, and then run a macro called ControlPanel.

A user form pops up in the spreadsheet and connects the curve-tracer force and measurement actions with the electronics (Figure 2). The possible operations are a single voltage measurement, a single forced-current output, or a sweep of current steps lasting 2.8 msec each and a voltage measurement at each step. The voltage measurements go into cells B4 to B10 in the spreadsheet. The resulting graph shows an x-y scatter plot of the data in cells A4 to B10. With this use of macros within Excel, all the graphing, analysis, and data storage common to Excel are still available to use. You can test the terminal behavior of many electronic components with this simple curve tracer. Resistors yield a linear plot whose slope is the resistor value (R=V/I). Diodes exhibit a nonlinear plot (I_D = I_0 + V/V_T). You can also plot a diode’s terminal behavior on a logarithmic current scale with a simple click within Excel’s charting capabilities. Some other application examples are forward-biased transistor junctions, LEDs, and relay coils.

The components in Figure 1 provide operation as low as 3V and low power consumption (low quiescent current). In the PWR block, resistor network R_N iso-lates and combines eight LPT1 outputs at D0 to D7 to power the circuit. The supervisory circuit, IC6, monitors the voltage from the LPT1 port. Use the LM3724 4.63V option for 5V PCs and the 3.08V option for 3.3V PCs. The reset output of IC6 goes back to the parallel port at terminal S5 for software-error checking and clears IC6 at start-up. IC6 also has a manual reset that provides direct user control. If you press momentary switch SW1, the output current resets to 1 μA. IC6, through IC3, Q_A, and IC5, and associated resis-tors R_A through R_M form a current-output D/A converter. Servoamplifier IC3, sets Q_A’s collector current. This current is a function of IC3’s reference voltage divided by the parallel combination of R_A to R_M. For the lowest current, 1 μA, only R_A connects (2.048V/2.16 kΩ). For the highest current (1 mA), all the resistors connect in parallel (2.048V/2.16 kΩ).

You select resistor values for a cumulative half-decade change in I_OUT steps by the square root of 10 in value. With only these seven resistors, the circuit covers three decades of current range. The pnp pair, Q_A and Q_AP, mirrors Q_A’s collector current to the terminal I_OUT. Emitter-degeneration resistors R_A and R_M improve the mirror’s output resistance. Shift register IC7 and open-drain inverter IC8 select which of the resistors to connect via program control. IC7’s input and clock connect the parallel port at C0 and C2 for serial shift-in operation. IC8’s on-resistance is lower than 40Ω. IC6 and IC8 perform voltage measurements (the ADC block). IC9, a rail-to-rail op amp buffers the lowpass filter comprising R_9 and C_5 at the parallel port at S6. IC9’s clock input provides timing control. When IC9’s chip-select input goes low, a conversion starts. Pulling I_OUT above the PC’s 5V level or below ground could result in circuit and PC damage. To be safe, operate this curve tracer with unpowered components.

The macros in the downloadable list- contain the basic interface features for changing the current-output values and measuring the voltage input. Within module 1, the declaration of Input32.dll must include its directory path. To minimize the effects of differing PC-clock and LPT1-bus speeds in different PCs, the user form performs a 10-sec timing cali-bration at initialization. This calibration attempts to set the I_OUT steps during a sweep to 2.8 msec. The software uses this time-delay coefficient throughout the program. Also, software-calibration coefficients within the code minimize voltage-measurement gain and offset errors. The spreadsheet maintains these coefficients for ease of changing. Use a voltmeter and a resistor of known value to calibrate. The initial gain coefficient in the spreadsheet for a 5V PC is 5V divided by 2^(3–1) (5/255=0.0196). The initial offset coefficient is zero. You can also calibrate the I_OUT current values in the spreadsheet. The user-form references these spreadsheet values. With external calibration, you can attain better than 1% error.
The need for a compact telemetry system poses a challenge for designing a small, light, low-component-count system. Interfacing serial data from the microprocessor is also difficult because most low-cost RF transmitters do not accept dc levels at the input. Commercial FSK (frequency-shift-keying) modulators are bulky and need many passive components. The circuit in Figure 1 uses a single NOT gate (inverter), an On Semiconductor NL27WZ14 in a surface-mount package, to generate continuous FSK data from TTL-level signals. The outputs from this circuit are compatible with available transmitters. When the TTL input has a low level, the circuit is a continuously running oscillator, producing approximately 2400 Hz (adjustable with R1). When the input assumes a high level, the oscillator’s frequency reduces by one-half with the introduction of a capacitor in the timing circuit via Q1. The inverter IC can accommodate an operating frequency of approximately 80 kHz. You can easily operate the FSK modulator at higher frequencies, such as 4800 and 9600 Hz, by reducing the values of the timing capacitors C1 and C2.