design ideas

Edited by Bill Travis

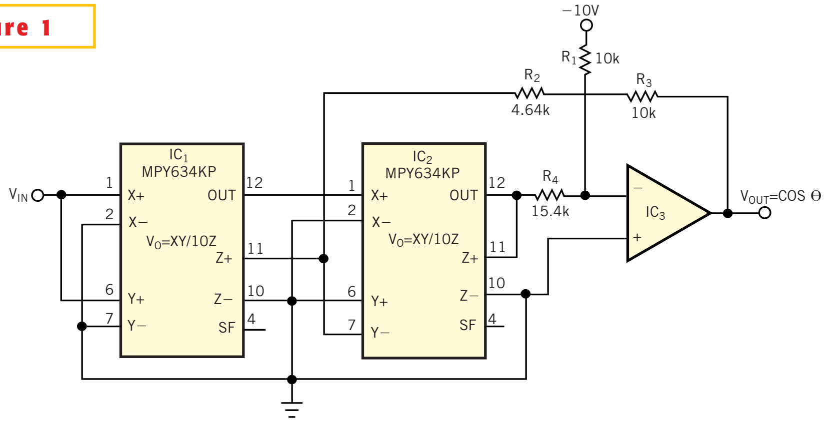
Circuit forms efficient cosine calculator

Matt Kornblum, Bradenton, FL

THE CIRCUIT in **Figure 1** converts a ±10V analog voltage representing an angle between θ_{MIN} and θ_{MAX} and emits a voltage equal to 10 cosθ. This circuit can have an accuracy of better than 1% over ±120° or better than 0.2% over ±90°. These figures represent an order-of-magnitude improvement over a Taylor-series estimate for the same range and for the same number of multiplications. The Taylor-series definition for a cosine (with θ in radians) is:

$$\mathrm{COS}\theta = 1-\frac{\theta^2}{2!}+\frac{\theta^4}{4!}\cdots+\frac{\theta^n}{n!}.$$

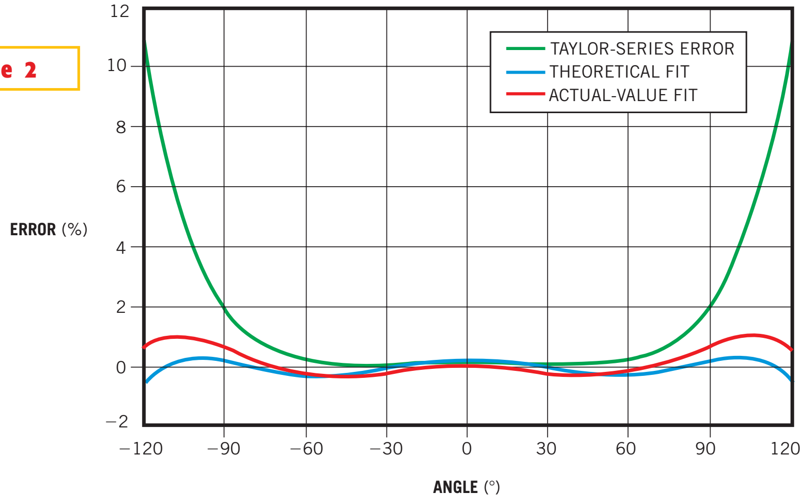
Figure 1



By manipulating Taylor-series coefficients, you can obtain better accuracy when generating cosines.

The series works well for high values of n or small angles. Generally, for n=4, significant errors start to accumulate for angles exceeding ±45°. When you use a Taylor-series expansion for better accuracies at larger angles, the number n becomes larger and demands more resources from the design. The Taylor series for n=4 has the form of $f(\theta)=a-b\theta^2+c\theta^4$, where a=1, b=0.5, and c=0.041667 (for angles in radians). By using a least-squares curve fit to optimize this function at n=4, you can find coefficients that allow you to obtain significantly better accuracies over the desired input range without raising the value of n to more than 4. The circuit in **Figure 1** embodies this least-squares approach.

Figure 2



For angles greater than ±90°, the revised coefficients in Figure 1 yield significant accuracy improvements in calculating cosines.

Choosing the resistor values for the circuit is relatively simple. Set R_1 and R_2 equal to each other (for 10V maximum input and a≈1), and determine values for R_2 and R_4 by applying the following equations:

$$R_2 = \frac{R_3}{b\theta^2_{MAX}},$$

and

$$R_4 = \frac{R_3}{c\theta^4_{MAX}}.$$

IC_1 generates the square of V_{IN} and negates it. This output sums through R_2 into IC_3. IC_2 generates the fourth power of V_{IN} and sums it into IC_3 through R_4. A −10V reference across R_1 creates the "a"-coefficient constant current into IC_3. The output of IC_3 is the sum of the three terms. Because IC_1 is an inverting amplifier, the circuit configures the multipliers such that the output of IC_1 is positive and the output of IC_2 is negative. Choosing the proper 0.1% resistors can improve circuit accuracy to better than 1% for −120 to +120°. You should use a low-offset op amp for best results. **Figure 2** shows the Taylor-series error, the theoretical fit, and the actual fit. For a fit in a 90° range, the values change slightly, and the errors across the range become significantly smaller. The constant "a" becomes 0.9996, b=0.4962, and c=0.0371. Then, $R_1=R_3=10\ k\Omega$, $R_2=8.16\ k\Omega$, and $R_4=44.2\ k\Omega$.

You can use the same approach to efficiently calculate cosine and sine values in a DSP system more rapidly than using a look-up table.

Is this the best Design Idea in this issue? Vote at www.ednmag.com.

Reference stabilizes exponential current

Tom Napier, North Wales, PA

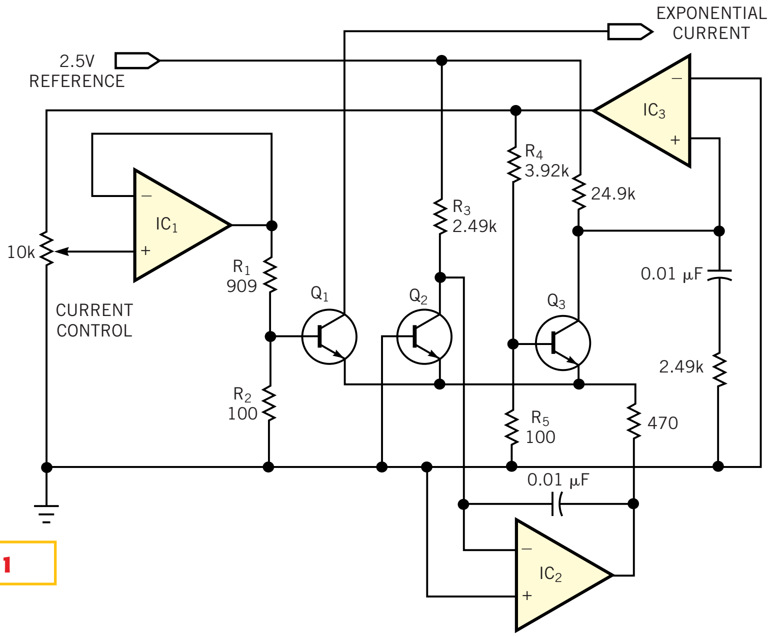


Figure 1

The use of a second reference eliminates temperature dependency in an antilog-ratio circuit.

In an antilog converter, the difference between the base voltages of two transistors sets the ratio of their collector currents:

$$I_1/I_2 = e^{V_{BE}q/kT}.$$

The use of matched transistors balances the first-order temperature coefficient but leaves a temperature-dependent gain term, q/kT. Classic antilog circuits use a thermistor in the drive circuitry to correct this temperature dependency. However, if the control input is a fraction of some reference voltage, as when you use a manual potentiometer or a DAC, you can achieve an exact temperature correction by adding a second reference transistor. **Figure 1** shows three of the five transistors in a CA3046 array. Q_1 is the exponential current source, and Q_2 is the conventional reference transistor. IC_2 forces Q_2's collector to ground so its collector current, 1 mA in this example, is simply the reference voltage divided by R_3. Typically, this current equals the maximum output required from Q_1; lower currents result from negatively driving the transistor's base.

The attenuator on the base of Q_1, R_1, and R_2 reduces the effects of IC_1's offset voltage. IC_3 drives the base of Q_3 via a second attenuator, R_4 and R_5, forcing its collector to ground. The reference current through Q_3 is a fraction of the main reference—one-tenth in this example. Despite the chip temperature, the base voltage of Q_3 is exactly the value you need to generate a 1-to-10 current ratio. Because IC_3's output supplies the reference voltage for the potentiometer, the ratio of the two attenuators defines the full-scale current-adjustment range. If the ratio is 4 to 1, the output current has a four-decade tuning range that's independent of temperature. The circuit in **Figure 1** is dynamically stable, using either low-power or fast op amps.

Is this the best Design Idea in this issue? Vote at www.ednmag.com.

Microcontroller becomes multifunctional

Abel Raynus, Armatron International, Melrose, MA

A microcontroller, by default, can execute only one program at a time. What do you do if, in a given project, you need to perform more than one operation at a time? Add more microcontrollers to the design? In certain cases it's unnecessary. Consider a real-life situation (**Figure 1**). The microcontroller constantly generates on its Pulse output pin a sequence of pulses with 25-msec duration and a repetition rate of 1 or 4 sec, depending on the state of the Rate input pin. LED illumination accompanies the pulse generation. Suppose that the microcontroller must simultaneously and independently perform some other functions using the rest of its six I/O pins. You can benefit from the fact that the pulse duration is much smaller than the repetition period. During this relatively long period, the microcontroller may not just wait for the generation of the next pulse, but, instead, it may perform some other operation. You organize the pulse-generating program as an interrupt-service routine and the rest of the program as a main program. To avoid any interference between these parts of the software, the interrupt-service routine execution time should be shorter than the smallest period of pulse repetition.

Listing 1 is the assembly routine for multifunctional operation. To make the interrupt program repeatable after the predetermined time interval, the best choice is to use the microcontroller's internal timer. This microcontroller has two timing options: timer-overflow interrupt and RTI (real-time interrupt). For a 2-MHz operating frequency, the timer overflow occurs every 0.51 msec.

LISTING 1—ROUTINE FOR MULTIFUNCTIONAL OPERATION

```
                        1  *********** Multifunctional operation   ***********
0000                    2  $include "std-j1a.asm" ;
0000                    3  $PAGEWIDTH 160
07F1                    4          org MOR;    resistor osc and input pulldown
07F1      20            5          fcb %00100000;
                        6  ******  I/O PORT BITS ****************************
07F2                    7  start   equ pA0
07F2                    8  rate    equ pA1
07F2                    9  pulse   equ pA4
07F2                   10  LED     equ pA5
                       11  ********** CONSTANTS ****************************
07F2                   12  N1      equ 16T  ;rate 65.5ms x16=1sec
07F2                   13  N2      equ 64T  ;rate 65.5ms x64=4sec
                       14  ********** VARIABLES ****************************
00C0                   15          org RAM
00C0                   16  N       rmb 1 ; time counter
                       17  ******     INITIALIZATION  **********************
0300                   18          org ROM
0300 [02] A630         19  init    lda    #%00110000 ;set I/O prtA
0302 [04] B704         20          sta    ddrA
0304 [05] 3F00         21          clr    prtA ;all output devices off
0306 [05] 3FC0         22          clr    N
0308 [02] 9A           23          cli          ;Interrupt enable
                       24  ***********************************************
0309 [05] 0100FD       25  MAIN:   brclr  start,prta,main ;wait for start
030C [05] 1808         26          bset   RTIE,TSCR       ;start pulse
generating
030E [05] 1A00         27          bset   LED,prtA        ;set LED on
0310 [02] 9D           28  work    nop  ;this instruction for illustration only
                       29          ; perform some other operations
                       30          ;
0311 [05] 0000FC       31          brset  start,prtA,work ;continue pulse
generating ?
0314 [05] 1908         32          bclr   RTIE,TSCR       ;stop pulse generating
0316 [05] 1B00         33          bclr   LED,prtA        ;set LED off
                       34          ;perform some other operations
                       35          ;
0318 [03] 20EF         36          bra    MAIN
                       37  *           Real Time Interrupt Service routine
                       38
*******************************************************
031A [05] 3CC0         39  RTI     inc    N                  ;N + 1
031C [03] B6C0         40        lda    N
031E [05] 030010       41        brclr  rate,prtA,rN1     ;if rate=0,go to rN1
0321 [03] B140         42          cmp    N2
0323 [03] 2509         43          blo    res
0325 [05] 3FC0         44  pls     clr    N
0327 [05] 1800         45          bset   pulse,prtA         ;generate one pulse
0329 [06] CD0337       46          jsr    dly25ms            ;  with 25 ms width
032C [05] 1900         47          bclr   pulse,prtA         ;
032E [05] 1408         48  res     bset   RTIFR,TSCR         ;RTIF reset
0330 [09] 80           49          rti    ;return from Real Time Interrupt
0331 [03] B110         50  rN1     cmp    N1
0333 [03] 25F9         51          blo    res
0335 [03] 20EE         52          bra    pls
                       53
*******************************************************
0337 [02] A620         54  dly25ms lda #32T       ;25ms delay
0339 [03] 5F           55  lp1     clrx
033A [03] 5A           56  lp2     decx
033B [03] 26FD         57          bne lp2
033D [03] 4A           58          deca
033E [03] 26F9         59          bne lp1
0340 [06] 81           60          rts
                       61  *********************************************
07F8                   62          org    VECTORS
07F8      031A         63          fdb    RTI
07FE                   64          org    VECTORS+6
07FE      0300         65          fdb    init
```

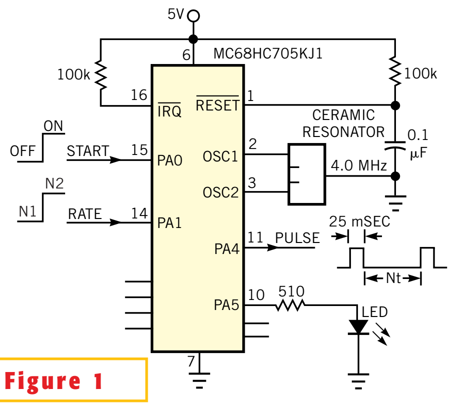


Figure 1

Between pulses, the microcontroller can perform other tasks using the software in Listing 1.

You can program the RTI period to be as long as 65.5 msec (with RT1-to-RT0=1-to-1). To simplify the counters, it is reasonable to choose the largest value: 65.5 msec. Then, to make the repetition periods equal to 1 and 4 sec, you create the counters modulo 16 and 62 accordingly in the RTI routine (**Listing 1**). You can see in **Listing 1** that the microcontroller waits for a high level on its Start pin to begin pulse generation and LED lighting. During the interval between pulses, it performs the other operations, continuously checking the state of its Start pin. After receiving a low level on the Start pin, the microcontroller stops pulse generating and switches off the LED. You can download the software for multifunctional operation from the Web version of this article at www.ednmag.com.

Is this the best Design Idea in this issue? Vote at www.ednmag.com.

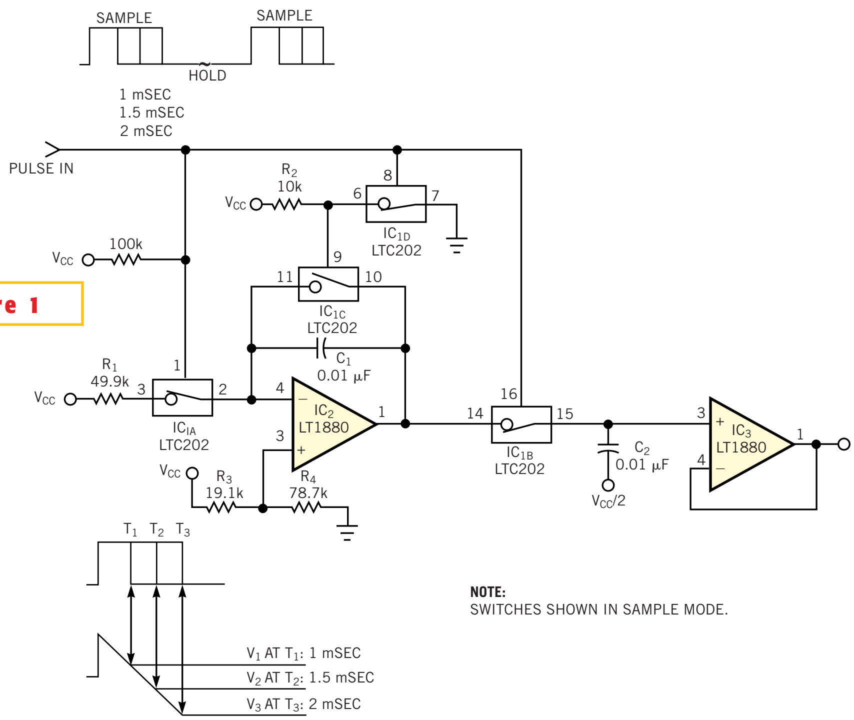
Circuit converts pulse width to voltage

James Mahoney, Linear Technology Corp, Milpitas, CA

The circuit in **Figure 1** converts pulse information to a clean dc voltage by the end of a single incoming pulse. In another technique, an RC filter can convert a PWM signal to an averaged dc voltage, but this method is slow in responding. Converting low-duty-cycle pulse information is slower yet. The circuit in **Figure 1** uses two low-input-bias-current LT1880 op amps, IC_2 and IC_3, and an LTC202 quad analog switch, IC_{1A}, IC_{1B}, IC_{1C}, and IC_{1D}, to configure the integrator and sample-and-hold stages that convert a single pulse to a dc voltage. The circuit's output is stable after a single pulse. This example shows the conversion of a low-duty-cycle positive pulse, whose width varies from 1 to 2 msec with a period of 25 msec, to a clean dc voltage. The input pulse starts, stops, and resets the integrator and controls the input to the sample-and-hold stage. After the reset operation, the positive pulse level-triggers the integrator, comprising R_1, C_1, and IC_2. The sample-and-hold stage, comprising IC_{1B}, C_2, and IC_3, is in the sample mode, sampling the output of the integrator, while the incoming pulse is high.

When the incoming pulse goes low, the circuit disconnects the input to the sample-and-hold stage, putting it into hold mode. The integrator then stays in the reset state until the next positive pulse arrives. During reset, analog switch IC_{1A} opens to disconnect the integrator's input, switch IC_{1C} closes to reset integration capacitor C_1, and switch IC_{1B} opens to disconnect the input to the sample-and-hold stage, placing the stage in hold mode. Analog switch IC_{1D} inverts the on/off states of switch IC_{1C}. The

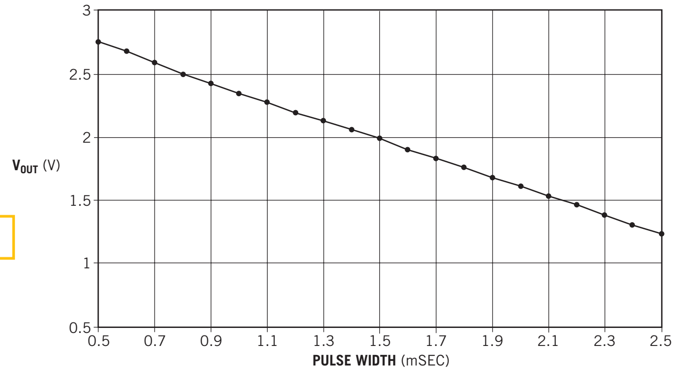
Figure 1



NOTE:
SWITCHES SHOWN IN SAMPLE MODE.

This circuit yields a clean dc voltage that indicates the width of an incoming pulse.

Figure 2



The circuit in Figure 1 linearly converts a pulse width to a dc voltage.

LT1880 op amp is a good choice for the integrator and sample-and-hold stages because of its maximum input-bias current of 900 pA at 25°C and maximum of 1500 pA maximum over the full −40 to +85°C ambient-temperature range. Another benefit of the LT1880 is its maximum input-offset-voltage drift of 1.2 µV/°C. Integrator capacitor C_1 and resistor R_1 set the conversion gain.

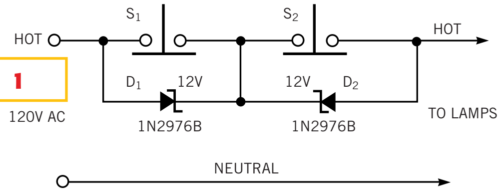
You should use polypropylene, polystyrene, or Teflon capacitors for C_1 and C_2 to minimize integrator drift and sample-and-hold droop rate. The voltage ratio that resistors R_3 and R_4 set establishes the dc level at the positive pulse's midrange value: 1.5 msec in this example. **Figure 2** shows input pulse width versus output voltage. You can easily modify the circuit in **Figure 1** to yield different conversion gains, output levels, and swings for different pulse widths. The circuit operates with pulse-width information and not duty-cycle values. The sample-and-hold stage is an analog-memory element that reveals the dc-voltage equivalent for this pulse width.

Is this the best Design Idea in this issue? Vote at www.ednmag.com.

Short dc power-line pulses afford remote control

Tom Hornak, Portola Valley, CA

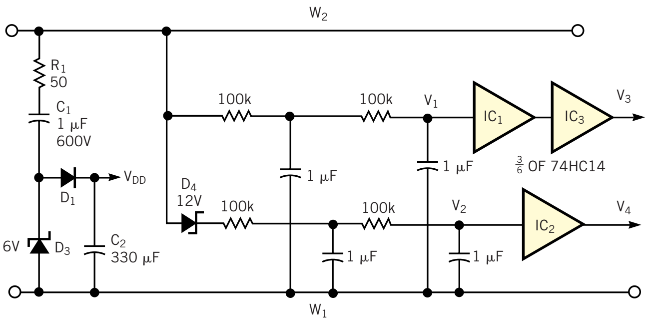
Figure 1



This circuit creates dc pulses for use with control circuitry located at the load.

If you face the challenge of adding a second, independently controlled light source to an existing ceiling lamp controlled by a wall switch, you may find that stringing a second power line is impossible. First, you can replace the wall switch by the circuit in **Figure 1**. Pushing the on switch S_1 or S_2 for approximately 1 sec inserts the 12V zener diodes D_1 or D_2 in series with the hot wire of the power line. During the push, the polarity-dependent conduction of the zener diodes creates a small positive (negative for D_2) dc component across the line and only slightly reduces the line's 120V-ac component. A control circuit at the lamps' site reacts selectively to the polarity of this dc pulse and controls the power to the two lamps. The required power rating of the two zener diodes depends on the load current. The short duration and low duty cycle of the activation are helpful. The 1N2976 diodes in **Figure 1** are rated for continuous dissipation of 10W.

Figure 2

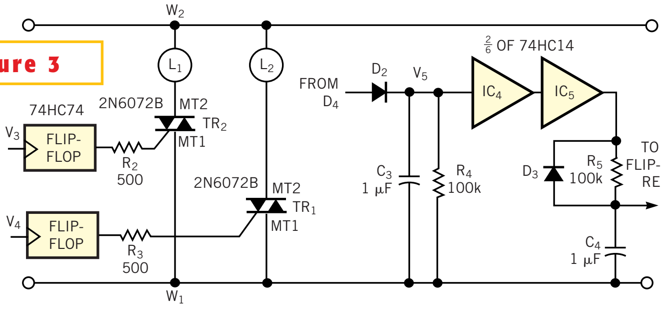


This control circuit uses dc pulses from the circuit in Figure 1 to drive triacs in the circuit in Figure 3.

Figure 2 shows the first part of the control circuit located at the lamps' site, including the two leads of the power line, W_1 and W_2. Current through capacitor C_1 and resistor R_1 creates a 60-Hz square wave across the 6V zener diode, D_3. Diode D_1 and filter capacitor C_2 generate a dc supply voltage of $V_{DD}=5V$ for the control circuit's active elements. Two two-stage RC filters connected to W_2 create V_1 and V_2, with reference to W_1. The filters attenuate the 120V-ac voltage between W_1 and W_2 to a subvolt level in V_1 and V_2. An extra zener diode, D_4, creates a positive 5V dc bias in V_2. The filter outputs V_1 and V_2 drive inverting Schmitt triggers IC_1 and IC_2. Inserting zener diode D_1 by pushing S_1 in **Figure 1** changes V_1 from 0V to 5V and V_2 from 5V to 10V. Inserting D_2 in **Figure 1** by pushing S_2 changes V_1 from 0V to −5V and V_2 from 5V to 0V. Note that the input-protection diodes of IC_1 and IC_2 limit the voltage swings of V_1 and V_2. The output V_3 of IC_3 responds to pushing S_1 by a positive transition and has no response to pushing S_2. The output V_4 of IC_2 responds to pushing S_2 by a positive transition and has no response to pushing S_1.

Figure 3 shows the second part of the control circuit located at the lamps' site. Signals V_3 and V_4 in **Figure 2** drive the clock input of toggle flip-flops IC_1 and IC_2, respectively. For clarity, **Figure 3** doesn't show the connections of the flip-flops of $\overline{Q}$ to D and the Set terminal to V_{DD}. When you push switch S_1 in **Figure 1**, the positive transition in V_3 toggles flip-flop IC_1. Similarly, when you push S_2, the positive transition in V_4 toggles flip-flop IC_2. Thus, you can independently control the states of flip-flops IC_1 and IC_2 by pushing S_1 and S_2, respectively. To drive the two lamps, the Q outputs of the flip-flops drive the gates of triacs TR_1 and TR_2 via coupling resistors R_2 and R_3. The MT2 terminal of each triac drives the lamps, L_1 and L_2, respectively. Pushing S_1 changes the state of lamp L_1; pushing S_2 changes the state of lamp L_2. Thus, you have independent control of both lamps on a single power line. In this application, you want to keep each lamp's terminals safely connected to the hot and neutral wires. Therefore, you make W_1 the hot wire and W_2 the neutral wire.

Figure 3



Triacs independently control two loads based on signals from a pair of wall switches.

With the control circuit, the state of the flip-flops becomes uncertain if, after an interruption, the ac power returns. This situation is unacceptable because the lamps could turn on and stay on for an uncontrollable length of time. Therefore, you add a power-up reset circuit (**Figure 3**). To guarantee a safe reset also for short interruptions, the reset circuit must quickly pull down the flip-flops' Reset terminal, which is independent of V_{DD}'s slowly dropping level. Diode D_2 (driven by the 60-Hz square wave across zener diode D_4), capacitor C_3, and resistor R_4 act as an auxiliary rectifier supplying voltage V_5. When power experiences an interruption, V_5 drops to 0V much faster than V_{DD}. V_5 drives the cascade of inverting Schmitt triggers IC_4 and IC_5, which then quickly pull down the flip-flops' Reset terminals via diode D_3. When power returns, the Reset terminals pull up slowly via resistor R_5. The R_5C_4 time constant guarantees that the flip-flops' Reset does not release before V_{DD} reaches its full value.

Note that you can use this Design Idea in other applications. For example, if you omit the circuit in **Figure 3,** the transitions in V_3 and V_4 can drive an up/down counter that can perform an auxiliary control function for a device that the ac line powers. If you insert an additional conventional switch in series with the circuit in **Figure 1**, it can turn on and off the power to the device. If the application requires control signals near ground level, wire W_1 should be the neutral lead of the power line, and W_2 should be the hot lead. However, make sure that the powered device is not an inductive load because it can short out the controlling dc pulses.

Is this the best Design Idea in this issue? Vote at www.ednmag.com.