EDITED BY MARTIN ROWE
AND FRAN GRANVILLE

designideas

READERS SOLVE DESIGN PROBLEMS

Eight-digit counter works with common anode or common cathode

Charaf Laissoub, Valeo Interior Controls, Créteil, France

A classical design for directly driving eight seven-segment LED displays requires as many as 15 I/O lines. Previous Design Ideas have described many approaches for using a maximum number of LEDs with a minimum number of I/O lines (**references 1** through **5**). The following idea reuses one of these approaches to drive a maximum number of seven-segment LED displays, and it may be useful in designing a low-component-count, low-power, and low-cost LED-display module for a 24-bit frequency meter, for example.

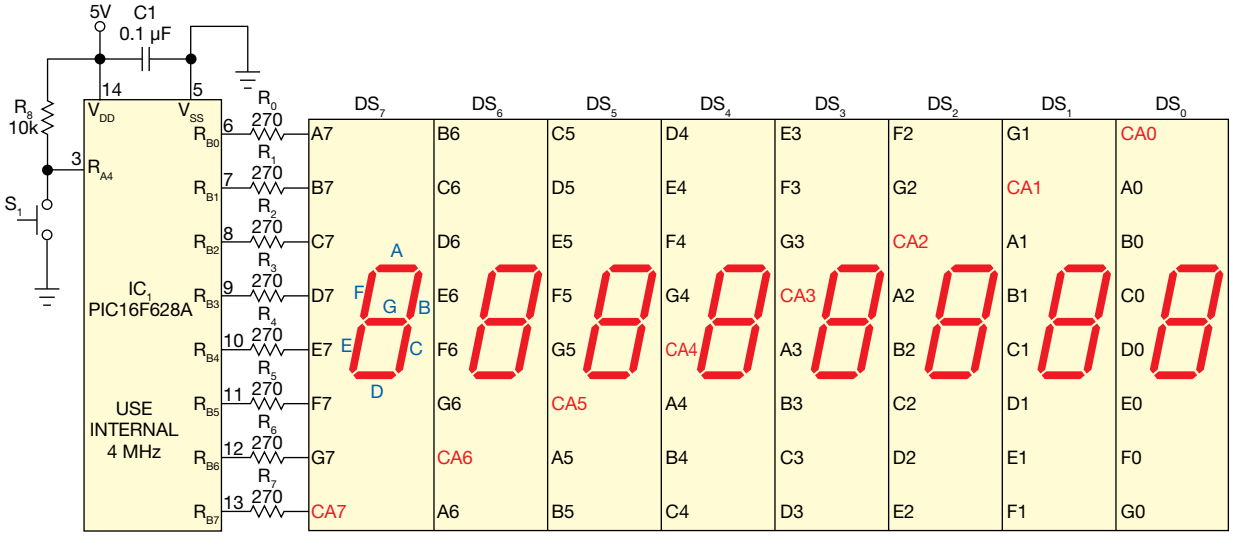
You can use the circuit in **Figure 1** to replace classical designs for digital counters that use TTL (transistor-transistor logic) or CMOS ICs. The single microcontroller is less expensive and readily available. By using conditional assembly in your programming, you can choose between common-anode and common-cathode configurations.

The algorithm uses double multiplexing, driving one digit at a time, segment by segment. This technique suits battery-powered designs because the circuit consumes a constant current of less than 2 mA when using superbright, seven-segment LED displays, such as KingBright's (www.kingbright.com) SC52-11EWA, and 270Ω resistors R_0 to R_7. Assembling the eight digits, DS_7, DS_6, DS_5, DS_4, DS_3, DS_2, DS_1, and DS_0, on a PCB (printed-circuit board) involves linking their corresponding pins A7, B6, C5, D4, E3, F2, G1, and CA0 to the I/O line, R_{B0}. **Figure 1** shows the connections.

This circuit uses the Microchip (www.microchip.com) PIC16F628A for test purposes. You can download assembly code at www.edn.com/100715dia and use it with any PICmicro midrange family, providing that a full 8-bit port is available. EDN

DIs Inside

▸To see all of *EDN*'s Design Ideas, visit www.edn.com/designideas.



NOTE: CHOOSE EIGHT COMMON-ANODE OR EIGHT COMMON-CATHODE, SEVEN-SEGMENT LED DISPLAYS AND UNCOMMENT ONE OF THESE TWO LINES IN THE INCLUDE FILE: "8dgtcacc.inc" #define use CAdisplay IF USING COMMON-ANODE, SEVEN-SEGMENT, DISPLAY OR #define use CCdisplay IF USING COMMON-CATHODE,SEVEN-SEGMENT DISPLAY.

Figure 1 This circuit can replace classical designs for digital counters that use TTL or CMOS ICs.

REFERENCES

1 Anonymous, "Microcontroller provides low-cost analog-to-digital conversion, drives seven-segment displays," *EDN*, May 10, 2007, pg 80, http://bit.ly/b3KBre.

2 Raynus, Abel, "Squeeze extra outputs from a pin-limited microcontroller," *EDN*, Aug 4, 2005, pg 96, http://bit.ly/aKKGEf.

3 Jayapal, R, "Microcontroller's single I/O-port line drives a bar-graph display," *EDN*, July 6, 2006, pg 90, http://bit.ly/barbMI.

4 Lekic, Nedjeljko, and Zoran Mijanovic, "Three microcontroller ports drive 12 LEDs," *EDN*, Dec 15, 2006, pg 67, http://bit.ly/deQTib.

5 Gadre, Dhananjay V, and Anurag Chugh, "Microcontroller drives logarithmic/linear dot/bar 20-LED display, *EDN*, Jan 18, 2007, pg 83, http://bit.ly/azQ9dw.

Count objects as they pass by

Vladimir Rentyuk, Zaporozhye, Ukraine

Counting objects is easy when the objects have regular motion, but it becomes more difficult when the objects vibrate or you have to manually move them. To deal with this problem, you need a reliable, error-free system of detecting an object. In this case, simple circuits with optical interruption switches, IR barriers, or other sensors don't work because objects may cross a sensor more than once.

The circuit in **Figure 1** solves this problem for objects such as cards, boxes, and even people. You can detect other objects if you use the proper sensors. The circuit produces two control pulses. One, OUT_1, occurs if any object runs through this system. It produces another pulse, OUT_4, only if an identified object passes through the system. These pulses let you count both the number of objects and the identified objects that pass through the system. The system doesn't produce counting errors if objects repeatedly cross each sensor, even when the objects return to the system, provided the object does not move from Sensor 1's zone. The system requires no difficult mechanical unit for stabilizing the speed of the moving objects. The counter selects objects that are slightly longer than the distance between its sensors. **Figure 2** shows how the circuit tracks an object between the sensors.

THE CIRCUIT PRODUCES TWO CONTROL PULSES.

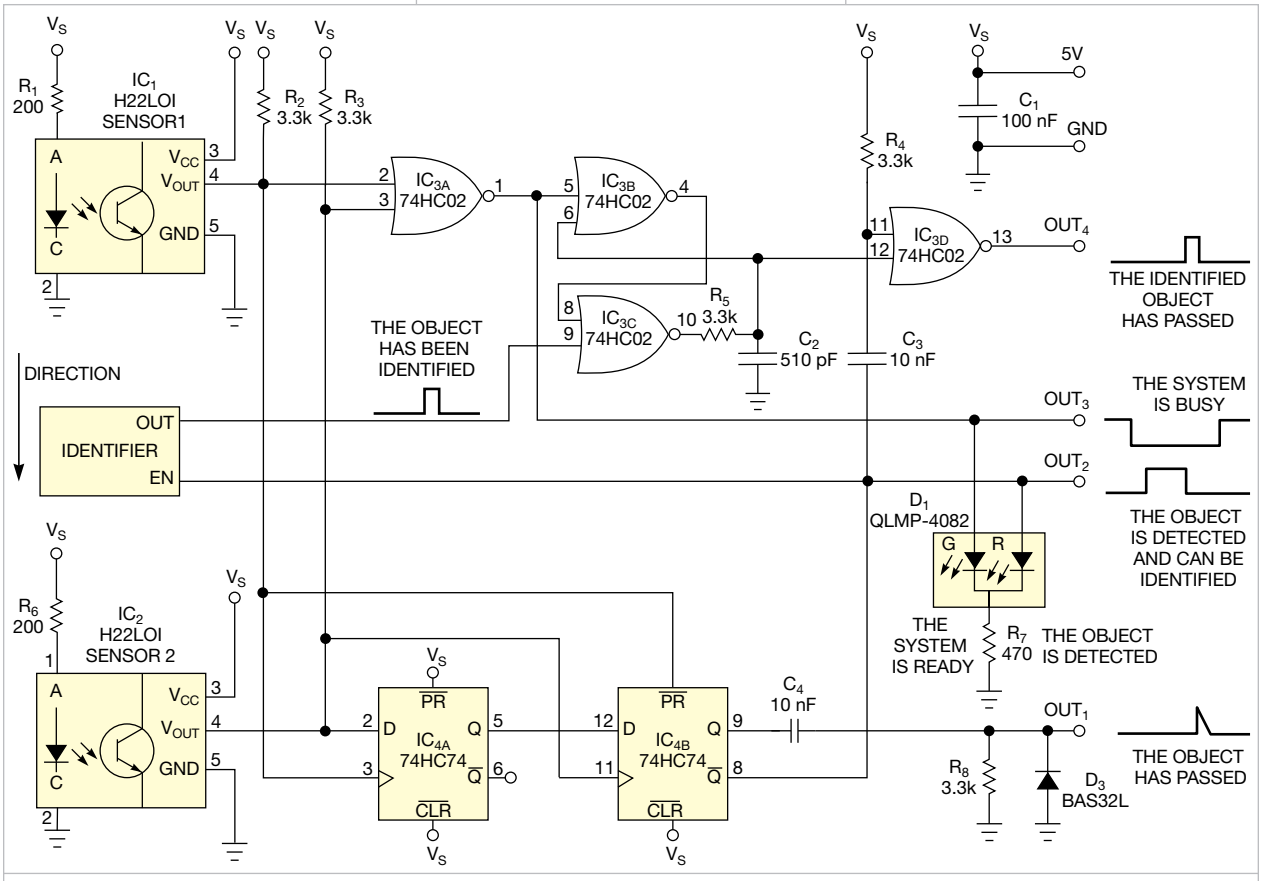


Figure 1 This circuit counts objects such as cards, boxes, and even people.

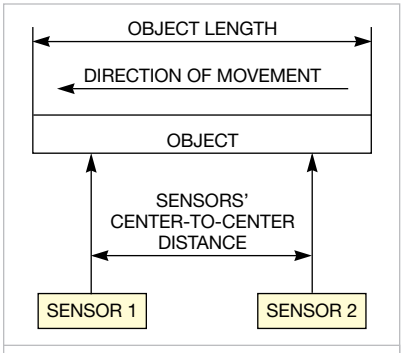


Figure 2 The circuit tracks movement using an object's length and direction.

The object counter comprises IC_1 and IC_2, H22LOI sensors from Fairchild Optoelectronics Group (www.fairchildsemi.com). The sensors are optical-interruption switches with open-collector outputs. A control circuit uses IC_4, a 74HC74 dual D-type flip-flop, and IC_3, a quad, two-input NOR gate. Sensors IC_1 and IC_2 produce logic-low output levels when objects are not between the IR diode and the IR receiver. LED D_1 shines green if the system is ready to take an object and red if the system has detected the object.

If the system is busy, LED D_1 does not illuminate. Resistor R_5 and capacitor C_2 protect RS trigger IC_{3B}/IC_{3C} from the chance of failure due to undesired signals. Some sensors require that you add buffers between them and the control circuit.

Table 1 describes the circuit's signals. A downloadable sheet describing the sequence of events is available at www.edn.com/100715dib. EDN

TABLE 1 SIGNAL DESCRIPTION

Trace or zone	Indication
Trace 1	OUT_1: common counting of passing objects
Trace 2	OUT_2: detection of an object
Trace 3	OUT_3: ready to take an object
Trace 4	OUT_4: counting an identified object
Zone A	Object identified
Zone B	Object not identified

Modified DDS functions as baud-rate generator

Vardan Antonyan, Glendale, CA

You can often use an available oscillator to generate a baud-rate clock for a UART. You must divide the oscillator frequency to attain the proper baud rate, but dividing can produce baud-rate errors. **Table 1** shows the percentage of error when you generate a baud rate using an 8-MHz crystal oscillator and a conventional binary divider. The system in this Design Idea obtains a clock 16 times faster than the baud rate.

Errors in baud-rate setting increase when the oscillator frequency doesn't match. In this case, you can add an oscillator operating at 18.432 MHz, for example, to minimize the error rate. Alternatively, you can use DDS (direct digital synthesis) to reduce errors at higher baud rates using the same oscillator (**Table 2**).

Reference 1 describes basic DDS operation. This design uses a simpler version of DDS with only a square-wave output (**Figure 1**). You can extract the square-wave output from the MSB of the phase accumulator. You can also add the divide-by-two stage to make the resulting signal with a 50% duty cycle. Calculate the baud-rate clock frequency using baud-rate clock=(reference clock×tuning word/2^N)/2, where N is the number of bits for the phase accumulator. A Verilog implementation of the DDS baud-rate generator using a 20-bit phase accumulator and 16-bit tuning word is available at www.edn.com/100715dic. EDN

TABLE 1 BAUD RATE WITH REGULAR DIVIDER

Baud rate	Divisor	Error (%)
50	10,000	0
300	1666	0.04
600	833	0.04
2400	208	0.16
4800	104	0.16
9600	52	0.16
19,200	26	0.16
38,400	13	0.16
57,600	8	7.84
115,200	4	7.84
230,400	2	7.84

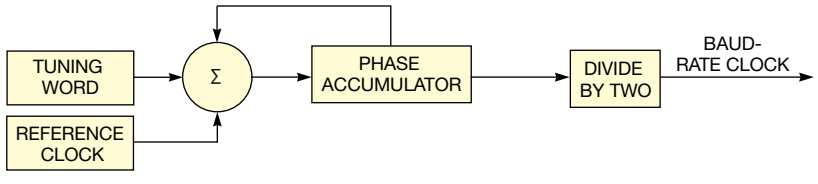


Figure 1 You can generate a baud-rate clock from an available oscillator.

REFERENCE

1 "A Technical Tutorial on Direct Digital Synthesis," Analog Devices, 1999, www.analog.com/static/imported-files/tutorials/450968421DDS_Tutorial_rev12-2-99.pdf.

TABLE 2 BAUD RATE WITH 20 DDS BITS

Output frequency	Phase word	Error (%)
50	13	–0.825
300	78	–0.825
600	157	–0.182
2400	629	–0.023
4800	1258	–0.023
9600	2516	–0.023
19,200	5033	–0.003
38,400	10,066	–0.003
57,600	15,099	–0.003
115,200	30,198	–0.003
230,400	60,397	–0.002

DC-voltage doubler reaches 96% power efficiency

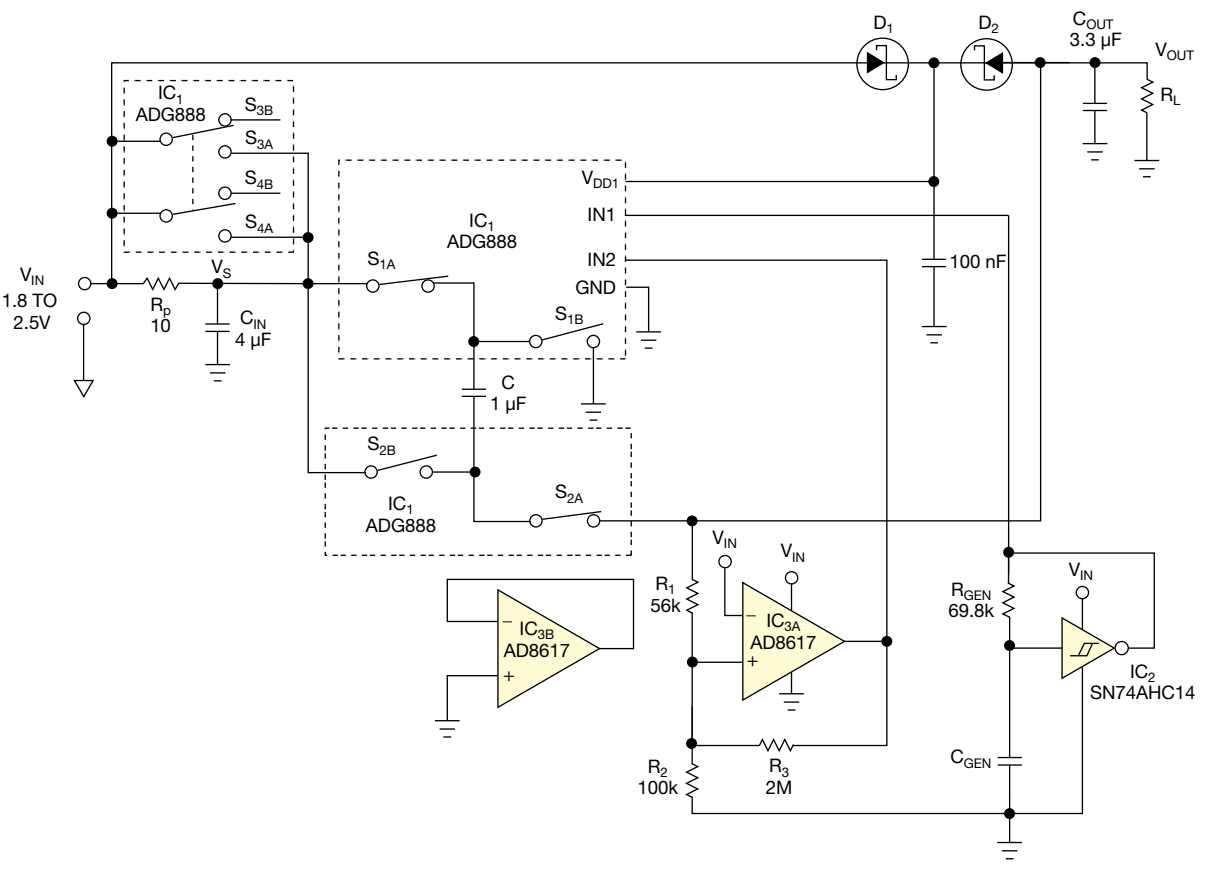
Marián Štofka, Slovak University of Technology, Bratislava, Slovakia

The voltage-doubler circuit in **Figure 1** can convert 2.5V dc to 5V dc or 1.8V to 3.3V. Most voltage doublers use an inductor, but this circuit doesn't need one. The circuit uses a capacitor, C, by charging it through serially connected switches. The charge switches let capacitor C charge, and the discharge switches are open. In the subsequent discharging phase, the charge switches are off, and the discharge switches close. The two discharge switches now connect capacitor C between the source of the input voltage, V_S, and the output capacitor, C_{OUT}. This connection scheme lets the applied voltages combine. Thus, the voltage at the output terminal has a value close to $2V_S$.

The two phases of operation repeat periodically at frequency f, which clock generator IC_2 determines. The duty cycle is about 50%, but the value isn't all that critical. One half of the Analog Devices (www.analog.com) high-performance ADG888 analog multiswitch provides the switching. The IC's two halves have independent control, so the other half occasionally shorts R_P, the 10Ω inrush-current-limiting resistor, which protects the charge switches from an initial overcurrent. That current occurs after power-on, before the output voltage reaches the predetermined percentage of the output's full voltage.

A micropower op amp, IC_{3A}, runs as a comparator with hysteresis. It compares input voltage to output voltage. Its output starts low and then goes high, which turns on paralleled switches S_3 and S_4. The comparator's action is ratiometric because the reference input voltage at the inverting input is the input-supply voltage, V_{IN}. This connection is possible because of the AD8617's rail-to-rail input/output operation. The circuit also provides overload protection for an excessive load, which connects to the circuit's output before power-on.

During soft start, the output voltage can't reach the threshold level for loads below a certain value. Consequently, the circuit remains in soft-start mode. The minimum value of R_L, which activates the protective subcircuit, is $R_L \le m^2 \times (\alpha/(1-\alpha)) \times R_P$, where the multiplication factor $m=(V_{OUT}/V_{IN})$ and α is



NOTE: C_{GEN} HAS AN OPTIMAL VALUE OF 62 pF WHEN R_L IS APPROXIMATELY 180Ω.

Figure 1 You can use this step-up dc/dc converter in applications in which power efficiency is a critical issue.

a fraction of V_{OUT}, at which the soft start turns off. For m=2, α=0.8, and R_P=10Ω, R_L is 160Ω. Thus, loads of 160Ω or less will overload the circuit if you connect them to the circuit's output before power-on. IC_2 and IC_3 get their power from the input supply. IC_1, however, switches voltages of as much as $2V_{IN}$, and its V_{DD1} supply-voltage pin must remain at the same level. An analog OR switch comprising Schottky barrier diodes D_1 and D_2 provides that voltage. The higher of the input or output voltages appears at the V_{DD1} pin of IC_1. The high levels of output voltages for both IC_2 and IC_3 suffice for control of IC_1 because the ADG888's data sheet allows a $0.36V_{DD1}$ value for the high value at the control inputs. The circuit has been tested at an input voltage of 2.386V, R_L of 178.46Ω, a frequency of 200 kHz, a supply voltage of 2.377V, an input supply current of 51.285 mA, and an output voltage of 4.588V. Evaluating these data gives a multiplication factor of 1.929 and power efficiency of 96.39%.

This power efficiency remains more than 96% for frequencies of 150 to 350 kHz. The 9-mV drop at the switch-shorted R_P at the given input current indicates that the on-resistance of the paralleled switches has a value of approximately 0.175Ω.EDN

Microcontroller's serial port measures pulse width

Vishwas Vaidya, Tata Motors Ltd, Pune, India

Many industrial and instrumentation systems need to measure the duration of pulse inputs, such as frequency from rotational-speed sensors, gating and strobe pulses from external systems, and PWM (pulse-width-modulated) inputs. Designers generally use on-chip timers and edge-driven interrupts for this purpose. If one of these components is unavailable, however, you can employ an unused on-chip serial-synchronous receiver to make those measurements.

You can set the baud rate of the serial-port receiver for the necessary timing accuracy. The receiver interrupts the microcontroller after every 8 bits. You can embed the pulse-width acquisition routine, which resides in your application program, to read the byte that the ISR (interrupt-service routine) receives. It counts and accumulates the number of ones and zeros the bytes receive to measure the duration of an incoming pulse (**Figure 1**).

The algorithm measures the duration between two consecutive rising edges.

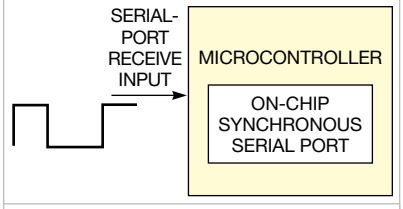


Figure 1 A microcontroller's serial port can receive pulses from various sources.

The microcontroller detects a rising edge or a falling edge when a received byte is neither 0xff nor 0x00. If a byte is less than 0x80H (100 000b), then the byte marks a rising edge. If the byte is equal to or greater than this value, then the byte marks a falling edge.

The leading rising edge resets the bit counter to the number of trailing ones in the rising-edge byte by shifting the bits to the right. The bit counter increments by eight at the arrival of every byte, including the one that marks the falling edge. When the counter receives the trailing rising edge, marked by the next rising-edge byte, which is greater than 0x80h but less than 00H, it again counts the number of leading zeros in this byte and adds them to the accumulated-bit counter. The accumulated-bit count at this point directly relates to the time period of the pulse train by a factor equaling the baud rate.

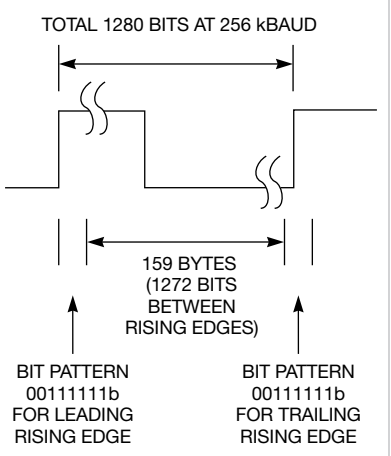


Figure 2 The algorithm uses a 001111 bit pattern to detect edges.

Figure 2 depicts a 200-Hz pulse train, which has a 5-msec period between two consecutive rising edges. The baud rate is 256 kbaud. During a measurement cycle, assume that the leading rising edge is marked as 0011 1111b. The microcontroller counts the number of trailing ones by shifting them right and initializing the bit counter as six. This count corresponds to approximately 23.43 µsec.

Next, every byte before the rising-edge byte increments the bit counter by eight. Simple calculation shows that the sum is 159 bytes, or 1272 bits. At this point, the total bit count is 1278, including six one bits received in the first rising-edge byte.

The pulse train now encounters its trailing rising-edge byte as 0011 1111b. When this encounter occurs, you need to shift the zeros left to count two bits. The total bit count between the rising edge now is 1280. At a 256-kbaud rate, this figure corresponds exactly to 5 msec, or 200 Hz.

Figure 3, a flow chart, is available with the online version of this Design Idea at www.edn.com/100715did. It explains how you can use this concept to measure frequencies in hundreds of hertz.

You can tailor this bit-counting concept to your application's requirements. For measuring only a low period of a pulse, you need to detect a falling edge and count the bits until you encounter a rising edge. You can use this concept to read an incoming PWM signal by reading high periods of a known incoming pulse frequency.EDN