Single-Wire Sensors Interface Directly To A PC

D.S. Oberoi — *CEDTI, Jammu, India*

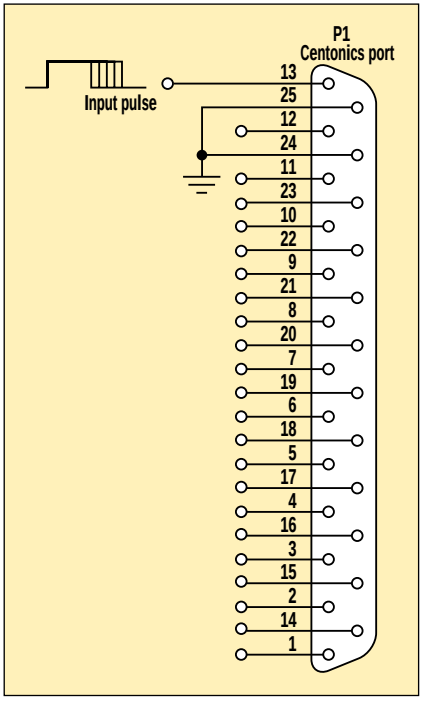
Harinder Dhingra — *GCET, Jammu, India*

Over the years, various vendors have developed a number of interfacing techniques to help address design requirements for simplified signal conditioning, reduced component count, and lowered development and system costs. One such low-cost interfacing technique is the use of pulse-width-modulated (PWM) digital outputs to eliminate the need for costly analog-to-digital converters (ADCs). This approach also decreases the number of signal wires required.

Using this PWM approach, the measured data's value is passed on as a variable-length pulse to the measurement and control system through a single-wire interface. The on-period (t_{ON}) of the pulse indicates the value of the measured quantity. Any change in the measured value will cause t_{ON} to vary. The measured data can be captured to a specified degree of resolution by measuring t_{ON}. Modern microcontrollers have an onboard timer/counter peripheral that can easily be used to measure the pulse width.

PWM-output devices (single-wire interface) can be interfaced directly to a PC as shown in the figure. In a PC, there are no unused timer/counters available for measuring pulse widths. However, the PC's counter2 (a 16-bit, 8254 timer/counter), normally utilized for the PC's speaker operation, can be used for measuring the pulse width of a PWM signal. This counter operates using a clock frequency of 1.1931817 MHz (TIMER_FREQUENCY) and can be enabled or disabled by setting bit-0 of port 61h to one or zero, respectively.

For this application, counter2 is operated in mode 2 (i.e., as rate generator) and is configured by loading its control register at port 043h with a control word value of 0b4h. Initially, counter2 is loaded with an ffffh count at port 042h (i.e., counter2's read/write I/O port) in two cycles. During operation, the input signal is sampled continuously and, at the first rising edge, counter2 is enabled by setting bit-0 of port 61h to one. When counter2 is enabled, its count follows an auto-decrement pattern of one count every 0.838095 µs (1/TIMER_FREQUENCY). As long as the input pulse is high, counter2's count (elapsed_count) is read back in two cycles from port 042h. At this juncture of operation, the elapsed count is checked to see if the time period of the pulse is greater than ~54.9 ms (this is the maximum time period that counter2 can measure). If the input pulse width is greater than this limit, the overflow counter is incremented and the counter2 count is automatically set to an ffffh count. With the trailing edge of the input pulse, counter2 is disabled by setting bit-0 of port 61h to zero.



Single-wire sensors can be acquired directly by a PC if the counter/timer associated with the PC speaker function is reallocated for use by this application's pulse-width measurement function.

The counter data is stored in a data array (count[i]) whose length depends on the pulse width. If the pulse width is less then 1 ms, the length of the data array is 20; otherwise, it is 1. This is necessary to minimize the error when measuring small-pulse-width signals. The 20 array elements are arranged in ascending order (averagedata( )) and the average of the 10 middle values is used in the calculation of pulse width. Pulse width (pulse_width) is calculated using the overflow counter and the counter's count data. It is then displayed. Before starting the next read cycle, counter2 is reset with a count of ffffh. This process continues until the user presses the 'Q' key, which terminates the program and restores the screen attributes.

The software listing, written in C (Turbo C++, version 3.0) and assembly, can be found at *www.PlanetEE.com* by following the Ideas for Design link. It was tested in the MS-DOS mode of Win95 on a 450-MHz Pentium II. The minimum pulse width that could be measured was ~23 µs. The software can be easily modified to determine t_{OFF} or to calculate the frequency of the input pulse stream.