design ideas

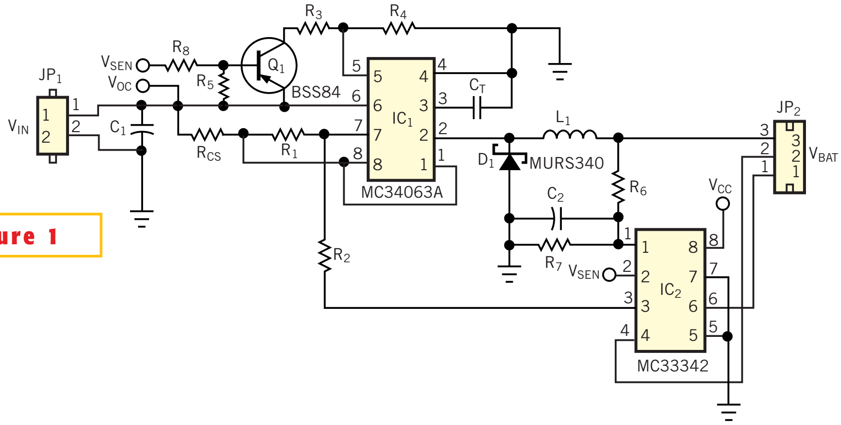
Edited by Bill Travis and Anne Watson Swager

Switching regulator charges NiMH batteries

Jason Hansen and Jim Hill, On Semiconductor, Phoenix, AZ

Many NiMH (nickel-metal-hydride) fast chargers use linear regulators, such as the LM317, in a current-source configuration with a charge-monitor IC. This arrangement dissipates much heat and requires considerable heat sinking, adding weight and cost. In the fast-paced, portable-system world, users do not want to carry heavy battery chargers that feel hot. A switching regulator can alleviate the weight and heat problem. In **Figure 1**, an MC34063A switching regulator and an MC33342 NiMH battery-charger IC combine to generate from an unregulated power supply a 600-mA battery charger for one to four NiMH cells. Using some tricks with the current limit and the feedback network, you can make a simple dc/dc converter into a programmable current source for NiMH charging. IC_1 operates in continuous mode in the buck converter without the output capacitor. The elimination of the output filter causes the inductor to act as the current source to charge the batteries. The externally programmable current limit sets the peak current in IC_1. You should select the inductor size so the peak-to-peak ripple does not exceed a predetermined level.

Figure 1



A switching regulator eliminates much of the heat in an NiMH battery charger.

The equation $V_1t_{ON} = V_2t_{OFF}$ helps determine the minimum input voltage to the system.

V_1 is the input voltage minus the saturation voltage of the switch, minus the battery voltage. V_2 is the battery voltage summed with the forward drop of the Schottky diode. In continuous mode, t_{ON} plus t_{OFF} is the total switching period. Rearranging the above equation for input voltage yields

$$V_{IN} = V_{SAT} + V_{BAT} + (V_{BAT} + V_D)\left(\frac{T}{t_{ON}} - 1\right).$$

The minimum input voltage to the system is 5.6V if t_{ON} is 7 sec, the switching frequency is 100 kHz, V_D is 0.3V, and V_{SAT} is 1.2V. Selecting 6V as the input voltage and allowing 0.1A ripple in the charging current and 1.4V across the inductor at maximum charge, the calculated inductor value is 126 µH. The R_6/R_7 ratio sets the fast-charge window. With V_{BAT} at 3.05V, slightly higher than the maximum charge level for the two cells, the ratio is 0.525. With 10-µA bias current through the resistive divider, R_6 is 27 kΩ, and R_7 is 51 kΩ. C_2 has a value of 1 nF and provides stability during fast-charge battery-voltage monitoring. C_1 provides input stability. Its size depends on the input-voltage ripple. C_T is 180 pF for 100 kHz. R_{CS}, 0.5Ω, is a function of peak current and the turn-off threshold current.

The feedback circuit of IC_1 connects to IC_2's V_{SEN} pin for the fast-charge monitor shutdown. During the fast charge, this open-collector output pin pulses low every 1.38 sec. R_3, R_4, R_5, R_8, and Q_1 make up the fast-charge monitor circuit. During normal-charge operation, Q_1 is off, and R_4 returns the feedback pin to ground. R_5 ensures that the base is high. When the MC33342 needs to measure the battery voltage, the V_{SEN} pin pulls the base of Q_1 low. The R_3/R_4 ratio ensures that the feedback pin is at a level higher than its threshold. You must understand the peak-voltage detector to properly design the dc/dc converter. The detector samples the battery voltage every 1.38 seconds. The sample time is 33 msec with an 11-msec preset time at the beginning. During this preset time, the MC33342 toggles the MC34063A into 0%-duty-cycle mode. This action shuts down the current to the battery so the MC33342 can take an accurate voltage reading. The inductor current must reach 0A during the initial preset time to minimize errors in the voltage reading.

R_{CS}, R_1, and R_2 make up the trickle-charge circuit. To effect trickle charge, this circuit connects to the current-limit pin of the MC34063A to create an offset for the maximum current. The open collector of the Fast/Trickle Charge pin goes low to enter trickle-charge mode. When the collector goes low, R_1 creates a voltage offset. R_1 and R_2 are much larger than R_{CS}; therefore, the offset current is very low. Because the current-sense pin monitors a low differential voltage across R_{CS}, V_{CC} needs to be within a few hundred millivolts of its nominal value for charge-current accuracy. The desired trickle-charge current determines R_1 and R_2. The target peak current for the trickle charge is 100 mA. The voltage across R_{CS} is 30 mV at the threshold; therefore, the voltage across R_1 must be 270 mV. For a current of 1 mA, the value of R_1 is 270Ω. Because the saturation voltage of the Fast/Trickle Charge pin is 0.2V, R_2's value is the input voltage minus the current-sense threshold voltage minus the saturation voltage of the charge-control pin divided by the selected current. The calculation yields approximately 5.5 kΩ for R_2. The component values in **Figure 1** produce a 600-mA battery charger.

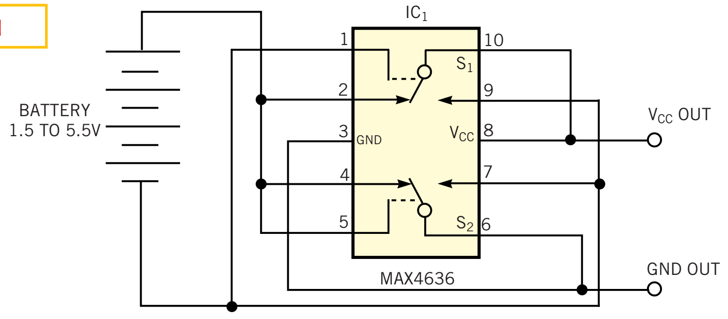
Is this the best Design Idea in this issue? Vote at www.ednmag.com/ednmag/vote.asp.

Circuit provides reverse-battery protection

John Guy, Maxim Integrated Products, Sunnyvale, CA

A universal problem in battery-operated devices is the threat of damage when an end user (never an engineer) inserts the battery backward. You can avoid damage by inserting a single diode or by using a diode-bridge configuration, but those fixes waste power and reduce the supply voltage by adding one or two diode drops between the battery and the supply rail. An alternative solution not only protects against battery-reversal damage, but also automatically corrects the reversal (**Figure 1**). To eliminate the voltage drops associated with discrete diodes, a low-on-resistance, DPDT (double-pole, double-throw) switch serves as a full-wave rectifier. When you insert the battery with the correct polarity as shown, the upper switch, S_1, is in its normally closed state, because its control pin is in its low state. The resulting connection from Pin 2 to Pin 10 provides a low-impedance path from the battery to the V_{CC} terminal. Conversely, the lower switch, S_2, closes its normally open terminal (not as shown) because its control pin is in its high state. The resulting path from Pin 7 to Pin 6 connects the battery's negative terminal to ground.

Figure 1



This circuit senses battery polarity, then quickly connects the load or swaps the battery leads.

The ESD-protection diodes in IC_1 guarantee start-up and act as a full-wave rectifier. MOSFETs internal to the analog switch turn on when the battery voltage exceeds 1V. Their less-than-20-nsec turn-on time enables the circuit to maintain normal operation by quickly swapping the leads of a reversed-polarity battery connection. The circuit resistance is proportional to the battery voltage. When the circuit operates from four NiCd, NiMH, or alkaline cells, the resistance in each leg of the rectifier is 2.5Ω (5Ω total). Operation with a two-cell battery (2.4 to 3V) yields a total resistance of 10Ω. IC_1 is rated for operation to 5.5V with 30-mA continuous current, making the circuit useful for cordless phones, portable audio equipment, handheld electronics, and other light- to medium-current applications. IC_1's miniature 10-pin µMAX package takes less space than four through-hole signal diodes and is almost as small as two SOT-23 dual signal diodes.

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

Transistor junctions monitor temperature zones

Matt Smith, Analog Devices, Limerick, Ireland

A simple circuit uses transistor junctions to monitor multiple temperature zones (**Figure 1**). The temperature sensors are ordinary, general-purpose, low-cost, diode-connected transistors. The well-known diode equation $V_{BE}=(kT/q)\times\ln(I_C/I_S)$ shows that there is a temperature dependency of approximately 2.2 mV/°C for a base-emitter junction. By forcing a two-level current through the base-emitter junction and measuring the resultant voltage, you can accurately determine the junction temperature, a technique known as ΔV_{BE} sensing. To prevent self-heating with this technique, current levels must be low. IC_2 uses this approach and supplies a low-level switched current source on its D+ and D− pins. An on-chip ADC converts the voltage information on D+ and D− into digital data that IC_2 stores in a register.

To monitor multiple-channel temperatures, you need to multiplex the measurement channels. A four-channel differential multiplexer, IC_1, selects the transistor junction that the circuit measures. The differential multiplexer ensures that D+ and D− remain as differential signals to preserve noise immunity. By cycling through the A_0 and A_1 address lines of the multiplexer, the µC or µP can poll each channel in sequence. If extra channels are necessary, you can add multiplexer channels.

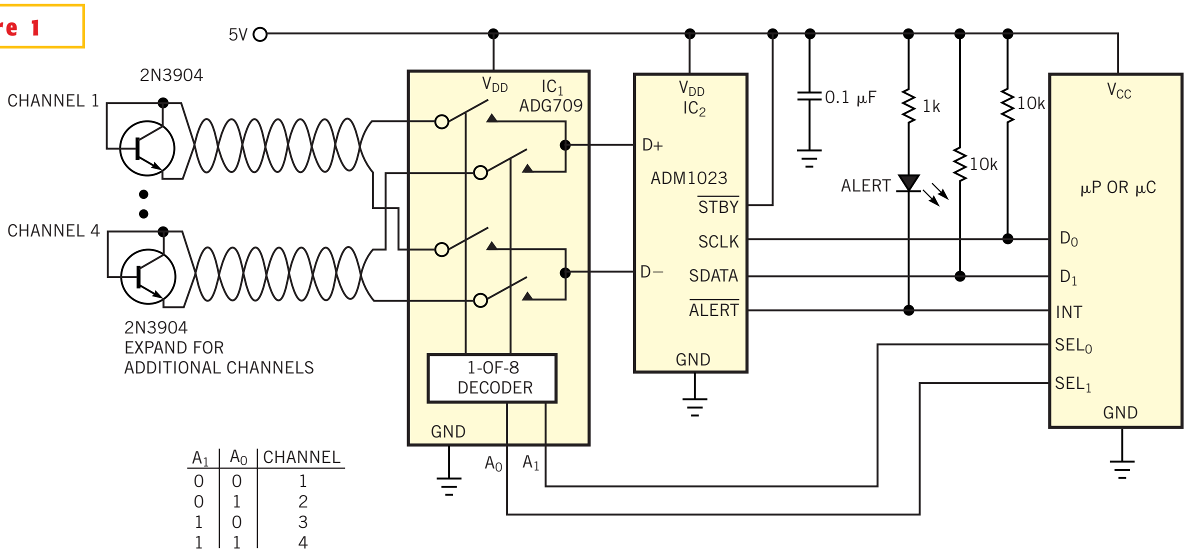
The on-resistance of the multiplexer channel results in a voltage drop across the channel. Therefore, you initially need to calibrate the circuit to remove this error. Fortunately, the error is constant because the channel resistance remains constant. You can use an offset register in IC_2 to store and automatically subtract the offset.

The remote-sensing transistors connect via a twisted-pair cable, and the cable can be as long as 50 ft. In extremely noisy environments, using a shielded twisted pair prevents the noise from interfering with the sensitive measurement. The circuit features a standard two-wire SMBus or I^2C interface, enabling communication with a µC or µP.

The circuit can accommodate a theoretical temperature range of −128 to +128°C. However, the practical range is more limited than these temperatures because moisture causes leakage currents and, hence, temperature errors. IC_2 also contains high- and low-limit registers and has an alert output. Thus, you can use the circuit to ensure that temperatures remain within an allowable band. Any deviation outside the limits, either high or low, results in activation of the alert output. The alert line drives an interrupt line on the µC or µP. The circuit can also detect fault conditions, such as open or short circuits, on the sensing elements. Fault conditions generate an alert signal. An interrupt-service routine can then interrogate the status register to provide fault identification and initiate corrective action.

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

Figure 1



A_1	A_0	CHANNEL
0	0	1
0	1	2
1	0	3
1	1	4

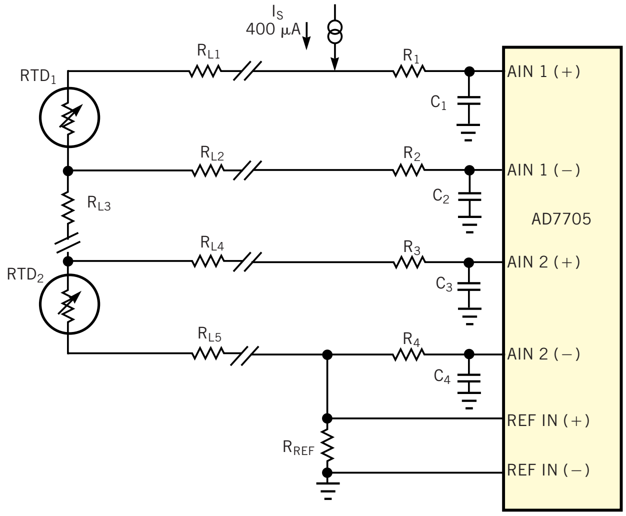
Diode-connected transistors monitor multiple temperature zones.

RTDs provide differential temperature measurement

John Wynne, Analog Devices Inc, Limerick, Ireland

You sometimes need to measure the differential temperature between two points in a system to a greater accuracy than that required in measuring the absolute temperature at either of the individual points. Differential-temperature measurement is necessary, for example, in monitoring heating-energy consumption in an apartment. Hot water enters from one pipe, circulates around the apartment through the radiators, and exits through a second pipe. Billing depends on the temperature differential between the entry and exit copper pipes, so absolute temperature is irrelevant. One way of measuring is to attach an RTD (resistance-temperature detector) to each pipe as it enters or exits the apartment and to take the voltage difference across the two RTDs. To ensure that the measurements are truly relative, you must wire the RTDs in series and excite them with the same current source, I_S (**Figure 1**). The same excitation current also flows through the reference resistor, R_{REF}, and generates the voltage reference for the ADC. Hence, the entire circuit is ratiometric. Therefore, both the current source, through, and the reference resistor need not be particularly stable over temperature for the circuit to operate properly. The circuit is also tolerant of ohmic drops in the connections to the three-wire RTDs.

Figure 1



This ratiometric differential-temperature measurement system eliminates ohmic wiring drops as sources of errors.

Channel 1 of the AD7705 reads an input voltage equal to $I_S R_{RTD1} + I_S R_{L1}$. Channel 2 reads an input voltage equal to $I_S R_{RTD} + I_S R_{L5}$. R_{L1} and R_{L5} represent the wiring resistances between the local electronics and the remote RTD elements. You should wire the RTDs such that $R_{L1} = R_{L5}$. Using software, subtract the ADC's Channel 1 reading from the Channel 2 reading. The ohmic drops cancel, leaving the differential temperature as the only remaining term. The inter-RTD wiring resistance, R_{L3}, does not appear in the equations and, therefore, has no effect. The input impedance of the AD7705 is very high, so essentially no current flows through R_{L2} and R_{L4}. The RC combinations act as lowpass filters that attenuate high-frequency noise that the wiring picks up. This filtering function is especially important with RTDs that are remote from the AD7705 and related measurement electronics. Choosing these components is straightforward, according to the data sheet at www.analog.com.

The RTDs give rise to a certain source of errors. Consider a common 100Ω platinum RTD with a resistance coefficient of 0.0038Ω/Ω/°C. This type of sensor, the European PRTD, is the most common RTD sensor. It is available in accuracy-tolerance classes A and B (or DIN A and DIN B), which specify both the initial accuracy at 0°C and the interchangeability over the operating range. Class A specifies $\pm(0.15+0.002|t|)$, and Class B specifies $\pm(0.3+0.005|t|)$, where t is the specified interchangeability temperature. You can buy two Class A, 100Ω, platinum RTDs from the same manufacturer and find that one is reading 0.2°C high at 25°C and the other is reading 0.2°C low at 25°C. This difference represents an apparent 0.4°C difference before you even commission the measurement system. To combat this initial error, you must either request a matched pair of RTDs from the manufacturer or calibrate out this difference at the time of installation. For instance, some sensor manufacturers sort PRTDs into tolerance groups with maximum Δt of ±0.05°C over 0 to 100°C. Alternatively, you can easily calibrate out the error by using the AD7705's separate gain and offset registers for the two channels.

The AD7705 specifies integral nonlinearity at 14 bits or better. However, the ADC measures the two inputs with 16 bits of peak-to-peak resolution. All this resolution is useful, because the ADC has the same linearity for either channel, whatever that resolution may be. This premise assumes that the gain of the ADC's internal PGA does not change between channels. Changing channels via the internal multiplexer does not contribute any additional error sources. Thus, differential-temperature measurements have a resolution of 14 bits or better.

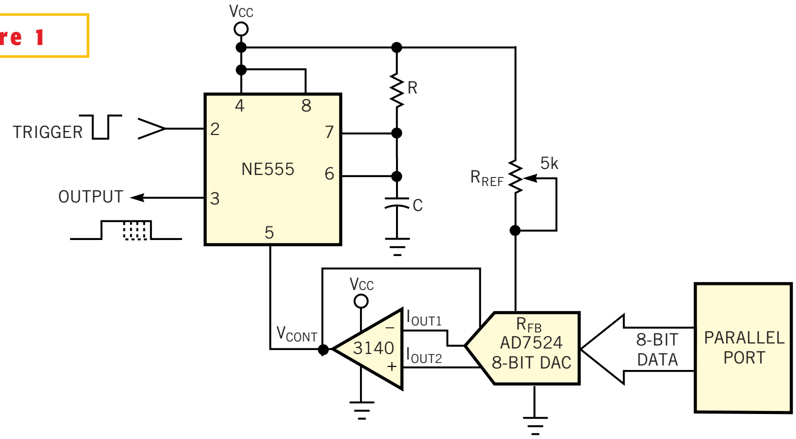
Is this the best Design Idea in this issue? Vote at www.ednmag.com/ednmag/vote.asp.

One-shot circuit is programmable

J Jayapandian, IGCAR, Tamil Nadu, India

Figure 1 shows how to digitally program the on-time of a one-shot multivibrator circuit. More and more, the Internet is playing a role in control operations in industrial and R&D endeavors and in household appliances. One-shot circuits are popular choices for the on/off control circuitry. You can interface the programmable one-shot design in **Figure 1** with any intelligent system, such as a PC, a µP, or a µC. The design uses a low-cost NE555 timer and an 8-bit AD7524 D/A converter. The timer IC is connected in a one-shot configuration with an on-time transfer function of t=1.1 RC. The control voltage on Pin 5 of the 555 can change the threshold of the comparator in the timer IC, thereby changing the on-time of the one-shot's output. In other words, by selecting the voltage on this pin, you can control the pulse width of the output waveform of the timer. The 8-bit DAC, with its MC 3104 op-amp buffer, provides programmable control of the one-shot's pulse width. You can send the required 8-bit word to the DAC via a PC's parallel port or from a µP or µC. The maximum on-time of the one-shot is a function of the R and C values. You can vary the pulse width from minimum to maximum by changing the bit pattern at the DAC's input from 00 to FF.

Figure 1



An 8-bit code applied to the DAC determines the on-time of this one-shot circuit.

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

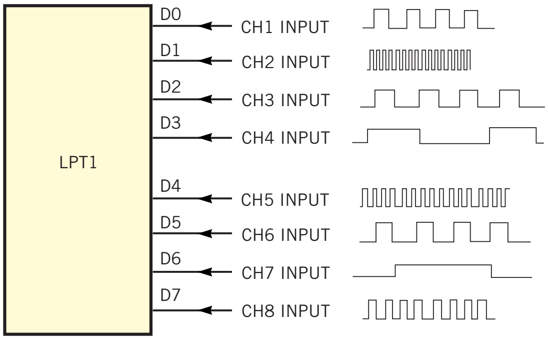
BIOS interrupt does eight-channel frequency counting

J Jayapandian, IGCAR, Tamil Nadu, India

The simple, no-cost connection in **Figure 1** provides frequency measurement for eight pulse trains through a PC's LPT1 port. The method uses a special BIOS interrupt and its handler routine without any hardware circuitry. The interrupt-handler routine, written in Turbo C, recognizes the external clock pulses coupled to the eight lines in the LPT1 port and determines their frequency, the number of pulses per second (**Listing 1**). The variable TICKER in the handler routine recognizes the occurrence of INT 1CH. The handler routine reads an 8-bit DATA word from the LPT1 port for every TICKER from the data received (the variable "COUNT"). The software extracts the 8 bits as eight-channel, single-bit data CH1, CH2, CH3, CH4, CH5, CH6, CH7, and CH8 by rotating right and by effecting a subsequent AND operation. For CH1 data, the 8-bit data received in "COUNT" is ANDed with 0x01, resulting in the first bit. For CH2, the "COUNT" data is ANDed with 0x02 and rotated right once. The remaining channel data undergoes an ANDing operation with 0x03, 0x04, and so on and rotation twice, thrice, and so on.

This method of converting data to a single bit for eight independent channels helps to monitor the variation in eight in-

Figure 1

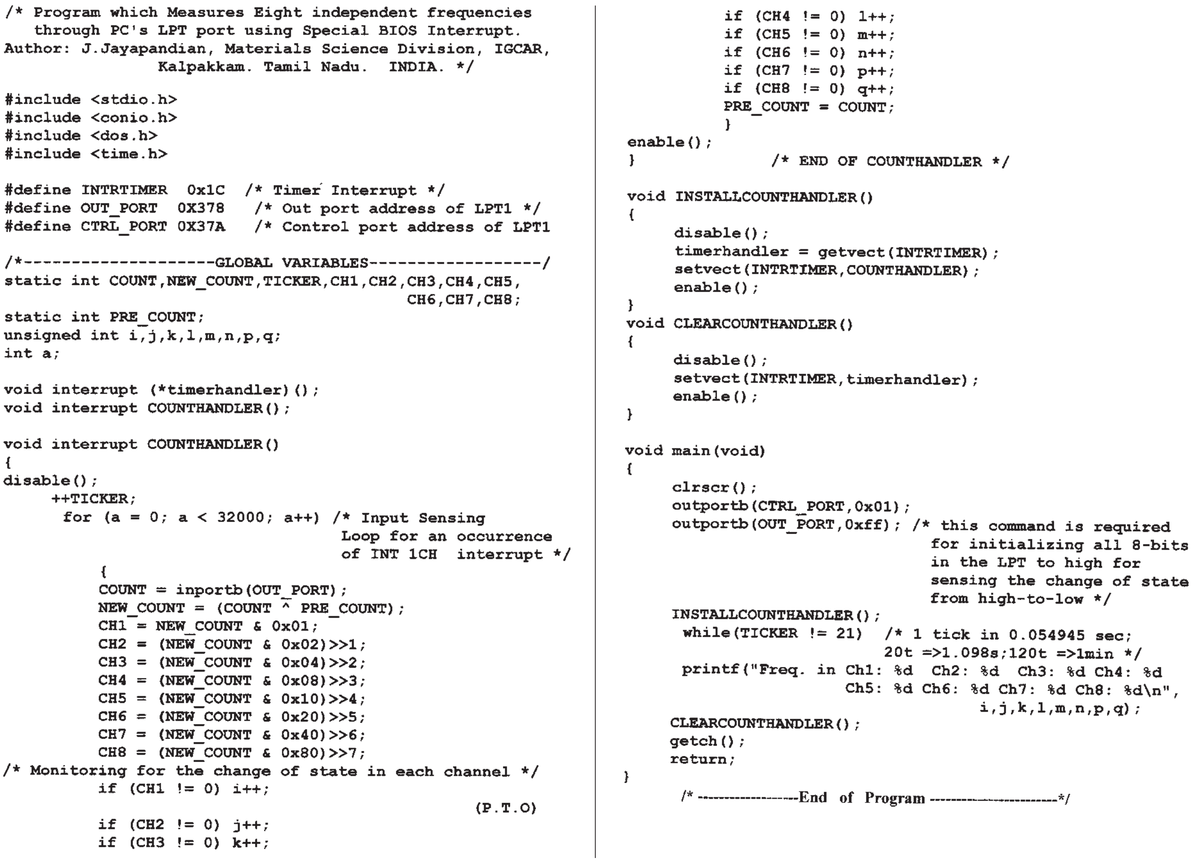


A PC's special BIOS interrupt measures eight independent frequencies.

dependent signals with respect to time. The handler routine immediately records any change in any one of the bits. The software increments variables I, j, k, l, m, n, p, and q if it detects a change in state of the bits in the corresponding channels 1 through 8. The completion of the 20th TICKER represents a time interval of 1 sec. The counts in variables I through p are a measure of frequency in channels 1 through 8, respectively. This method allows frequency measurement to 100 kHz. In the software example in **Listing 1**, the routine acquires 32,000 samples ("COUNT" from the LPT1 port); hence, for 1 sec and 20 TICKERs it is possible to acquire 6×10^5 samples with a 200-MHz Pentium system. The sampling variable, a, should not exceed the time of occurrence of INT 1CH. In this example, a is 32,000. You can download **Listing 1** from *EDN*'s Web site, www.ednmag.com. Click on "Search Databases" and then enter the Software Center to download the file for Design Idea #2620.

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

LISTING 1—BIOS-INTERRUPT HANDLER ROUTINE



```
/* Program which Measures Eight independent frequencies
   through PC's LPT port using Special BIOS Interrupt.
Author: J.Jayapandian, Materials Science Division, IGCAR,
                Kalpakkam. Tamil Nadu.  INDIA. */

#include <stdio.h>
#include <conio.h>
#include <dos.h>
#include <time.h>

#define INTRTIMER  0x1C  /* Timer Interrupt */
#define OUT_PORT  0X378   /* Out port address of LPT1 */
#define CTRL_PORT 0X37A   /* Control port address of LPT1

/*--------------------GLOBAL VARIABLES------------------/
static int COUNT,NEW_COUNT,TICKER,CH1,CH2,CH3,CH4,CH5,
                                          CH6,CH7,CH8;
static int PRE_COUNT;
unsigned int i,j,k,l,m,n,p,q;
int a;

void interrupt (*timerhandler)();
void interrupt COUNTHANDLER();

void interrupt COUNTHANDLER()
{
disable();
     ++TICKER;
      for (a = 0; a < 32000; a++) /* Input Sensing
                                     Loop for an occurrence
                                     of INT 1CH  interrupt */
          {
          COUNT = inportb(OUT_PORT);
          NEW_COUNT = (COUNT ^ PRE_COUNT);
          CH1 = NEW_COUNT & 0x01;
          CH2 = (NEW_COUNT & 0x02)>>1;
          CH3 = (NEW_COUNT & 0x04)>>2;
          CH4 = (NEW_COUNT & 0x08)>>3;
          CH5 = (NEW_COUNT & 0x10)>>4;
          CH6 = (NEW_COUNT & 0x20)>>5;
          CH7 = (NEW_COUNT & 0x40)>>6;
          CH8 = (NEW_COUNT & 0x80)>>7;
/* Monitoring for the change of state in each channel */
          if (CH1 != 0) i++;
                                                    (P.T.O)
          if (CH2 != 0) j++;
          if (CH3 != 0) k++;
          if (CH4 != 0) l++;
          if (CH5 != 0) m++;
          if (CH6 != 0) n++;
          if (CH7 != 0) p++;
          if (CH8 != 0) q++;
          PRE_COUNT = COUNT;
          }
enable();
}              /* END OF COUNTHANDLER */

void INSTALLCOUNTHANDLER()
{
     disable();
     timerhandler = getvect(INTRTIMER);
     setvect(INTRTIMER,COUNTHANDLER);
     enable();
}
void CLEARCOUNTHANDLER()
{
     disable();
     setvect(INTRTIMER,timerhandler);
     enable();
}

void main(void)
{
     clrscr();
     outportb(CTRL_PORT,0x01);
     outportb(OUT_PORT,0xff); /* this command is required
                                for initializing all 8-bits
                                in the LPT to high for
                                sensing the change of state
                                from high-to-low */
     INSTALLCOUNTHANDLER();
      while(TICKER != 21)  /* 1 tick in 0.054945 sec;
                              20t =>1.098s;120t =>1min */
      printf("Freq. in Ch1: %d  Ch2: %d  Ch3: %d Ch4: %d
                        Ch5: %d Ch6: %d Ch7: %d Ch8: %d\n",
                                         i,j,k,l,m,n,p,q);
     CLEARCOUNTHANDLER();
     getch();
     return;
}
      /* ------------------End  of  Program -----------------------*/
```