


# designideas

READERS SOLVE DESIGN PROBLEMS

## Flexible Hopfield neural-network ADCs quash noise

Paul J Rose, PhD, Mental Automation, Renton, WA

 A Hopfield network can convert analog signals into digital format and can perform associative recalling, signal estimation, and combinatorial optimization similar to the way a human retina performs first-level signal processing. This Design Idea explores the Hopfield-neural-network paradigm for ADCs.

Simple converters comprise one-layer neurons that accept analog inputs and generate digital-bit outputs; such neurons make up one form of adaptive- and distributive-processing networks. These neurons comprise voltage comparators driving either analog inverters or followers and fully connected feedback resistors from the analog outputs of the inverters or followers to the comparators (figures 1 and 2). Reference and analog-input voltages drive the neural networks, and digital outputs come from the comparators in the networks. Hopfield networks have learning capabilities; the circuit in this Design Idea can apply different adap-

tive-learning rules by using alternative comparator-inverter/comparator-follower schemes, conductance-node-layout schemes—reciprocals of the feedback resistances—between the input comparators, and bit-order readouts.

As the analog-input voltage increases, the circuit can produce either monotonically increasing (from a comparator-inverter scheme) or decreasing (from a comparator-follower scheme) bit-word outputs. Decreasing outputs are the complements of increasing outputs and suggest subtractive-bit operations. Further, you can shape the digital responses of the converters to analog-input voltages in varying degrees using different conductance-node layouts as part of rule adaptation. For further flexibility, reversing bit order for digital readouts allows for reflection of circuit responses about analog-input/digital-output characteristics.

You can simply state a few symbols and their meanings to construct the two converters. For energy functions,

### DIs Inside

62 8-bit microcontroller implements digital lowpass filter

64 Automotive switching regulators get input-transient-voltage protection

► What are your design problems and solutions? Publish them here and receive \$150! Send your Design Ideas to [edndesignideas@reedbusiness.com](mailto:edndesignideas@reedbusiness.com).

the resistive network conductances—synapse weightings ( $S$ ) in the form of reciprocal resistances ( $R$ )—have the designations  $S_{IJ} = 1/R_{IJ}$ , where  $I$  is the  $I$ th input comparator,  $J$  is the  $J$ th feedback path to the  $I$ th comparator, and  $I$  does not equal  $J$ —that is, there is no self-feedback path of the comparator to itself. The conductance between the input terminal of the  $I$ th comparator and the reference voltage,  $V_R$ , has the designation  $S_{IR} = 1/R_{IR}$ . The conductance between the input terminal of the  $I$ th comparator and the analog-input-signal voltage,  $V_S$ , has the designation  $S_{IS} = 1/R_{IS}$ .

For graphical curve fittings,  $Y$  is the normalized output-bit variable, and  $X$  is the normalized input analog voltage from a nonzero average value (less than one) to one.  $A$ ,  $B$ , and  $C$  are curve-fitting constants in the curve equation  $Y = 1 - A \times (1 - X)^C$  and the complementary-curve equation  $Y = A \times (1 - X)^C$ , where  $A$  is a coefficient,  $B$  is the lower limit for  $X$  and is less than one, and  $C$  is a power con-

**TABLE 1 INPUT VOLTAGE VERSUS OUTPUT WORD**

Input analog voltage (V)			Output binary word	
Raw range	Normalized range	Average normalized range	Raw	Normalized
0 to 0.189	0 to 0.2855	0.1427	0	0
0.189 to 0.265	0.2855 to 0.4003	0.3429	1000	0.5333
0.265 to 0.378	0.4003 to 0.571	0.4856	1100	0.8
0.378 to 0.662	0.571 to 1	0.7855	1110	0.933
More than 0.662	1	1	1111	1

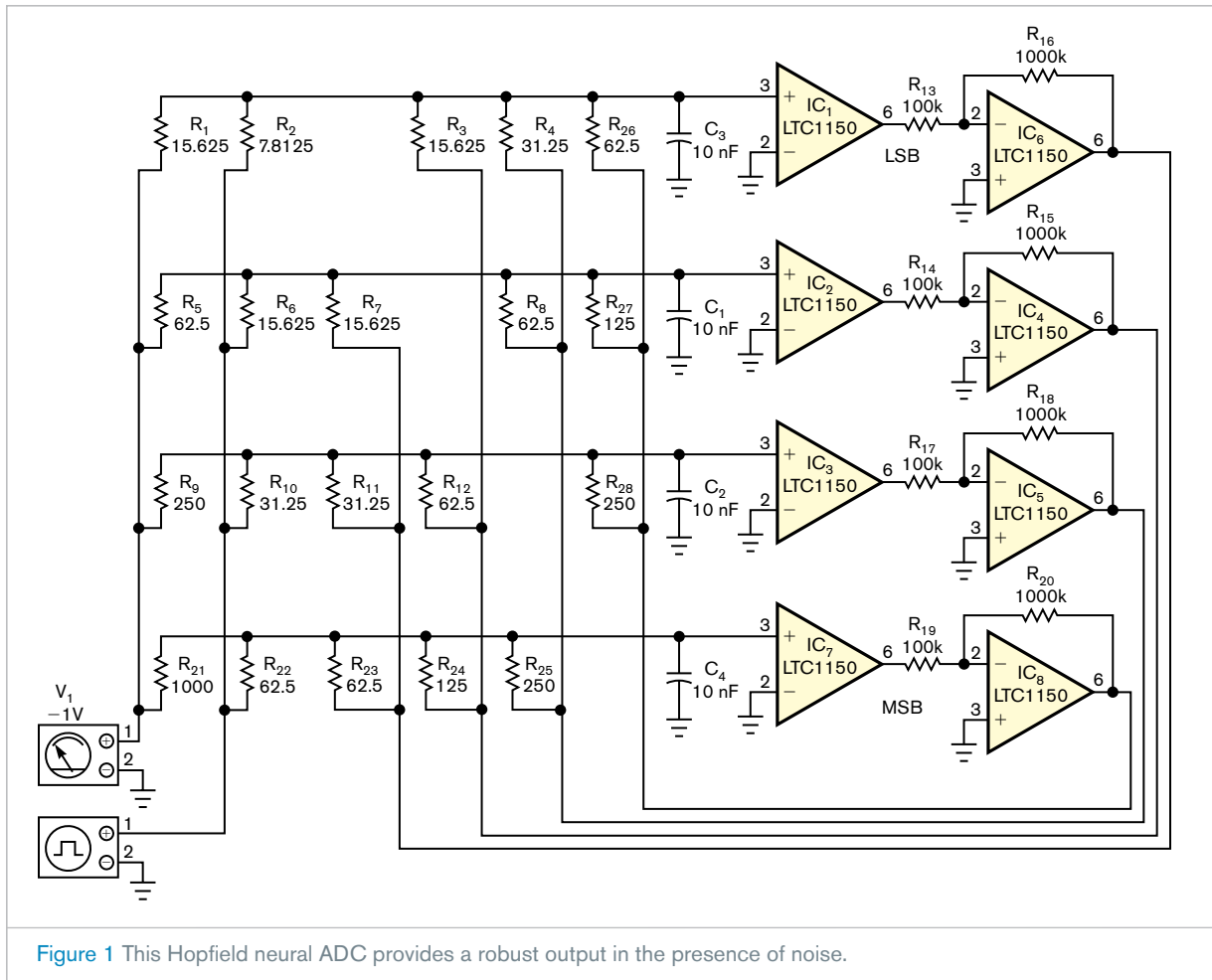


Figure 1 This Hopfield neural ADC provides a robust output in the presence of noise.

stant. For bit-pattern readout reversals, you can have the curve equation  $Y=A \times (X-B)^C$  and the complementary-curve equation  $Y=1-A \times (X-B)^C$ .

Figure 1 shows a 4-bit neural ADC employing voltage inverters that comparators feed. The comparators connect with their positive terminals joined to input nodes and with their negative terminals grounded. The bases of this network are inverse factors of one-half—that is, reciprocal factors of two—input-node conductances  $S_{ij} = -1 \times 2^{(2-1-j)}$ , where the  $-1$  factor comes from negative feedback through the related resistor;  $S_{IR} = 2^{(1-2 \times i)}$ ; and  $S_{IS} = 2^{(1-i)}$ . To determine node resistances, choose a maximum node resistance of  $1000\Omega$  corresponding to a minimum conductance of  $0.0078125$ , and a minimum node resistance of  $7.8125\Omega$  corresponding to a maximum conductance of one. Calculate all other

resistances from the ratios between the extremes of conductances. Using these values, you can construct Table 1. The table lists bits ranging from the most significant bit to the least significant. The table shows that the digitization process is inaccurate in that it is not linear with input voltage and with many intermediate bit words missing. But the process is precise because it is repeatable over sizable input-voltage ranges. From the table, you can derive the following curve-fitting equation:  $Y=1-1.6243 \times (1-X)^{3.1508}$ . When  $X$  is over the normalized range of  $0.1427$  to  $1$ ,  $A=1.6243$ ,  $B=0.1427$ , and  $C=3.1508$ . The  $Y$  equation is essentially cubic, and it quantitatively shows the highly nonlinear nature of the digitization process. You can obtain a “flipped” mirror—that is, not a true mirror, or pseudoscopic—version of the curve of the straight line on a

normalized graph by reversing the bit-order readout from the circuit so that the resulting curve equation would be:  $Y=1.6243 \times (X-0.1427)^{3.1508}$ .

Without analog-input-voltage transformation, such as the use of look-up tables or logarithmic amplifiers to process the input voltage, or digital corrective logic, digital responses from simple Hopfield neural converters are nonlinear and crude. However, these responses are still possibly useful for such applications as associative memory and pattern classification because of robustness in output precision.

Indeed, because of output digital stability, the Hopfield neural converter can allow for unwanted analog-input-signal noisiness or variations. This scenario is in strong contrast to conventional interface circuits between analog-transmission media and digital-

computing machines. This Design Idea shows that flexible circuit adaptability can exist in producing various forms of stable digital outputs from neural

ADCs depending on a designer's needs for neural-network-signal processing. This adaptability can be in the forms of various input-node-conductance

layouts; comparator/inverter and comparator/follower combinations; and the selected order of bit-readout patterns from the comparators. **EDN**

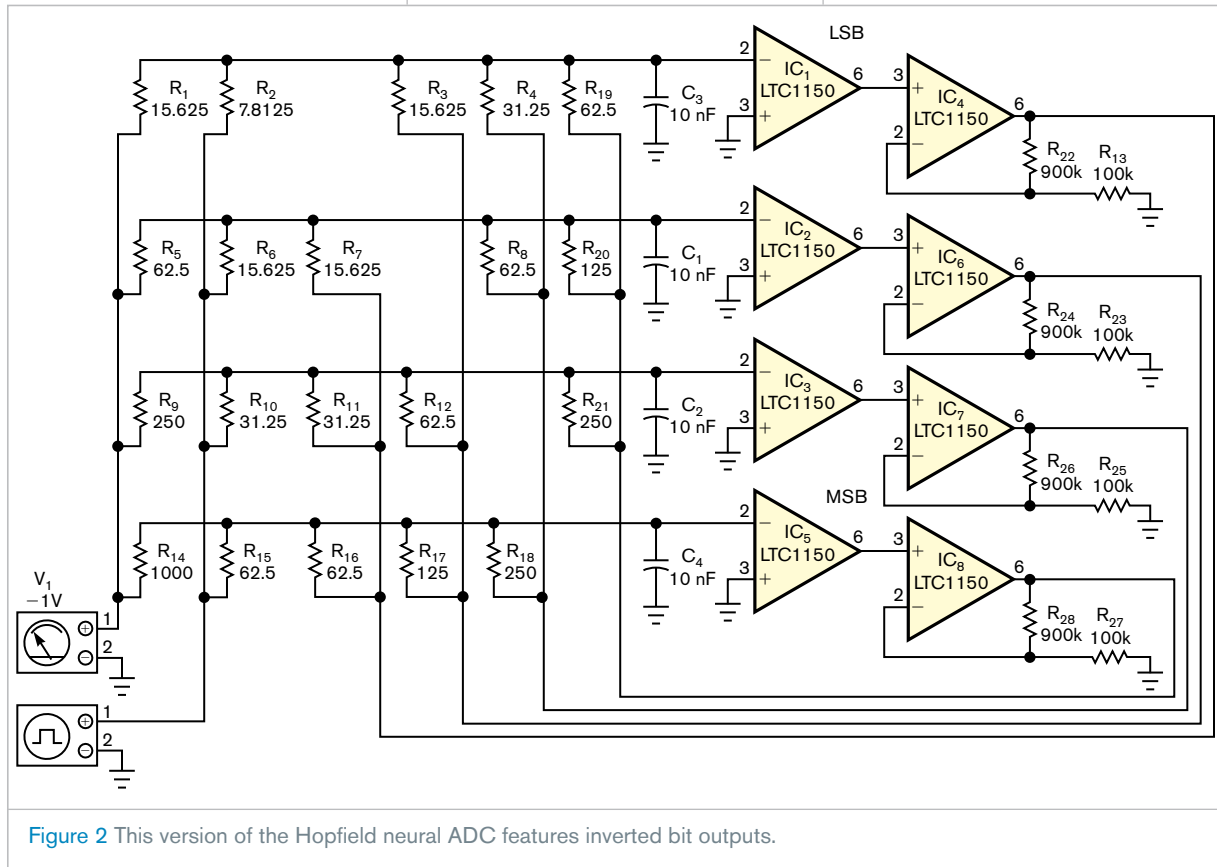


Figure 2 This version of the Hopfield neural ADC features inverted bit outputs.

## 8-bit microcontroller implements digital lowpass filter

Abel Raynus, Amatron International, Malden, MA

Filtering occurs frequently in the analog world. Unfortunately, in the digital world, engineers apply it mainly to the DSPs (digital-signal processors) and not to the small 8-bit microcontrollers that designers commonly use. This situation occurs because the math for the filter design is more complicated than most engineers are willing to deal with. Moreover, digital filtering requires calculations on integers instead of on floating-point numbers. This scenario causes two prob-

lems. First, the rounding-off error from the limited number of bits can degrade the filter response or even make it unstable. Second, you must handle the fractional values with integer math.

Several ways exist to solve these issues. For example, you can use operations with 16-, 32-, and 64-bit numbers, or you can scale for better accuracy. These and other methods usually require more memory, and, as a result, the program often does not fit into a small microcontroller. A literature

search shows that published digital-filter firmware is written in C. Programs in C need more memory than those written in assembler. This situation often makes them unacceptable for small microcontrollers with limited memory resources.

**Listing 1**, available at the Web version of this Design Idea at [www.edn.com/080124di1](http://www.edn.com/080124di1), shows a simple engineering method to design single-pole, lowpass-digital-filter firmware for 8-bit microcontrollers. The low-end Freescale ([www.freescale.com](http://www.freescale.com)) MC68HC908QT2 is the target of the assembler program, but you can apply this Design Idea to any type of microcontroller because it uses only standard assembler instructions.

Leaving aside the sophisticated design methods based on Z transformation with its extensive math, this idea uses another approach based on a recursive equation. You calculate each output-signal sample as the sum of the input signal and the previous output signal with corresponding coefficients. A recursive equation defines a single-pole lowpass filter as:  $Y[n]=X[n]\times a_0+Y[n-1]\times b_1$ , where  $X[n]$  and  $Y[n]$  are input and output values of sample  $[n]$ ,  $Y[n-1]$  is an output value of the previous sample  $[n-1]$ , and  $a_0$  and  $b_1$  are weight coefficients that decrement  $\delta$  controls. The coefficients have the value of  $0<\delta<1$ ,  $a_0=1-\delta$ , and  $b_1=\delta$ . Physically,  $\delta$  is the amount of decay between adjacent output samples when the input signal drops from a high level to a low level. You can directly specify the value of  $\delta$  or find it from the desired time constant of the filter,  $d$ , which is the number of samples it takes the output to rise to 63.2% of the steady-state level for a lowpass filter. A fixed relationship exists between  $d$  and  $\delta$ :  $\delta=e^{-1/d}$ , where  $e$  is the base of natural logarithms. The preceding equations yield  $Y[n]=Y[n-1]+(1-\delta)\times(X[n]-Y[n-1])$ .

## NUMERICALLY PERFORMING THE FILTERING FUNCTION PROVIDES THE BENEFIT OF CONSISTENCY BECAUSE COMPONENT TOLERANCES, TEMPERATURE DRIFT, AND AGING DO NOT AFFECT THE FILTER'S ALGORITHM.

Instead of multiplying a decimal-point number,  $1-\delta$ , it is more convenient for assembler programming to divide by the reciprocal integer,  $F=1/(1-\delta)$ :  $Y[n]=Y[n-1]+(X[n]-Y[n-1])/F$ . Thus, you can determine the digital filter's parameters using the following steps:

1. Choose the parameter  $F$ . For assembler, it is convenient to perform division as right shifts. For right shifts, the value of  $F$  should be  $2^S$ , where  $S$  is the number of shifts. Let  $F$  equal 8, which you reach after three right shifts.

2. Calculate the decrement:  $\delta=1-1/F=1-1/8=0.875$ .


3. Calculate the time constant as  $d=-1/\ln\delta=-1/\ln 0.875=7.49$  samples.

The equation  $Y[n]=Y[n-1]+(X[n]-Y[n-1])/F$  determines the design of the microcontroller's algorithm for the filter. The algorithm needs three registers: input for  $X[n]$ , output for  $Y[n]$ , and an increment register to keep the  $(X[n]-Y[n-1])/F$  term. The size of these registers depends on the inputs. In this application, the signals from the built-in 8-bit ADC range from 00 to \$FF and must go through the lowpass filter. So, the input and the output registers are 1 byte in size. To increase the accuracy of division, add half the divisor to the dividend. This action increases the increment register to 2 bytes.

Numerically performing the filtering function provides the benefit of consistency because component tolerances, temperature drift, and aging do not affect the filter's algorithm. The implementation of the digital filter in the microcontroller gives the additional benefit of flexibility to adjust the filter's parameters, because this flexibility depends only on the firmware. **EDN**

## Automotive switching regulators get input-transient-voltage protection

Kevin Daugherty, National Semiconductor, Novi, MI

 Engineers often face difficult trade-offs when voltage regulators can encounter high-voltage transients that are well above normal input-supply operating ranges. This situation is common in automotive applications in which high-voltage transients from an alternator load dump can produce transients of 36 to 75V for durations as long as 400 msec. Designers must choose between a regulator that can withstand such maximum input voltage or use an input-protection scheme. The simple circuit in this Design Idea provides a highly cost-effective method for clamping an input voltage from a battery input with transients as high as 50V to take advantage of a 20V, 3-MHz regulator. With this

circuit, your design can achieve a small total footprint with relatively low cost because of the 3-MHz operation along with lower voltage components than might otherwise be necessary to withstand 50V.

Input-protection components consist of  $Q_1$ ,  $R_1$ ,  $D_1$ ,  $C_5$ , and one-half of  $D_2$  (Figure 1). At start-up, N-channel MOSFET  $Q_1$ 's source is at ground potential and turns on when  $R_1$  applies the battery voltage to the gate. Once the input voltage is above the minimum of 2.74V on  $IC_1$ , the LM2734Z regulator starts switching, which charges the bootstrap circuit comprising  $D_3$ ,  $D_4$ , and  $C_B$ . This bootstrap voltage of approximately  $V_{OUT}-V_{FD}$  (forward-voltage drop) of  $D_3$  then transfers to

the gate source of  $Q_1$ . Capacitor  $C_5$  then maintains gate drive during the bootstrap diode's off times.

Under normal operating conditions, for example, the battery voltage is 8 to 18V,  $D_1$  does not limit conduction of  $Q_1$ , and the gate voltage tracks approximately 2.5V above the input-supply voltage for a low voltage drop from the battery voltage to the input voltage of the LM2734Z. However, when the input voltage increases above the threshold that  $D_1$  sets, the input voltage to the LM2734Z regulates to the zener voltage ( $V_Z$ ) of  $D_1$  minus the threshold voltage of  $Q_1$ , or approximately  $20-2V=18V$ , well below the 24V absolute maximum of the LM2734Z. Selecting  $Q_1$  requires careful consideration of maximum input voltage, gate-to-source-voltage threshold, and power dissipation under both steady-state and thermal-transient conditions.

$Q_1$ , the SI1470DN N-channel MOSFET, provides 50V protection

with a drain-to-source voltage ( $V_{DS}$ ) of 30V+20V (zener diode  $D_1$  voltage), has an on-resistance of 95 m $\Omega$  at a gate-to-source voltage of 2.5V, and comes in a thermally efficient SC70-6 package. For some applications, the

regulator's output voltage may be insufficient to fully turn on the selected protection MOSFET, so you can increase the bootstrap voltage with a separate zener reference, as the LM2734Z's data sheet shows (**Reference 1**).**EDN**

## REFERENCE

1 "LM2734Z Thin SOT23 1A Load Step-Down DC-DC Regulator," National Semiconductor, [www.national.com/pf/LM/LM2734Z.html](http://www.national.com/pf/LM/LM2734Z.html).

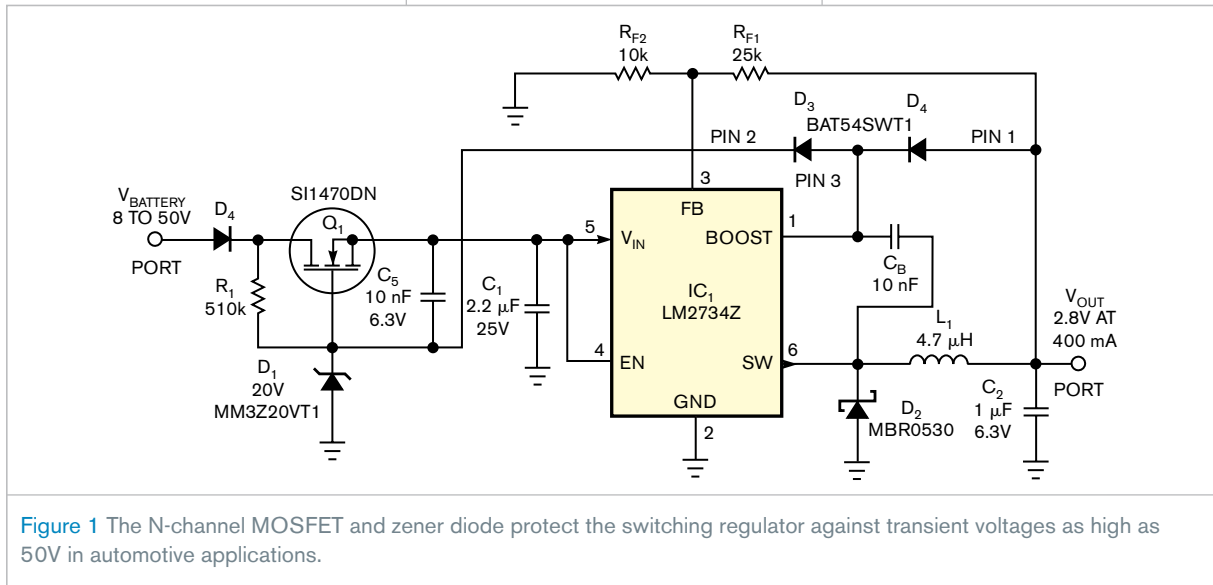


Figure 1 The N-channel MOSFET and zener diode protect the switching regulator against transient voltages as high as 50V in automotive applications.