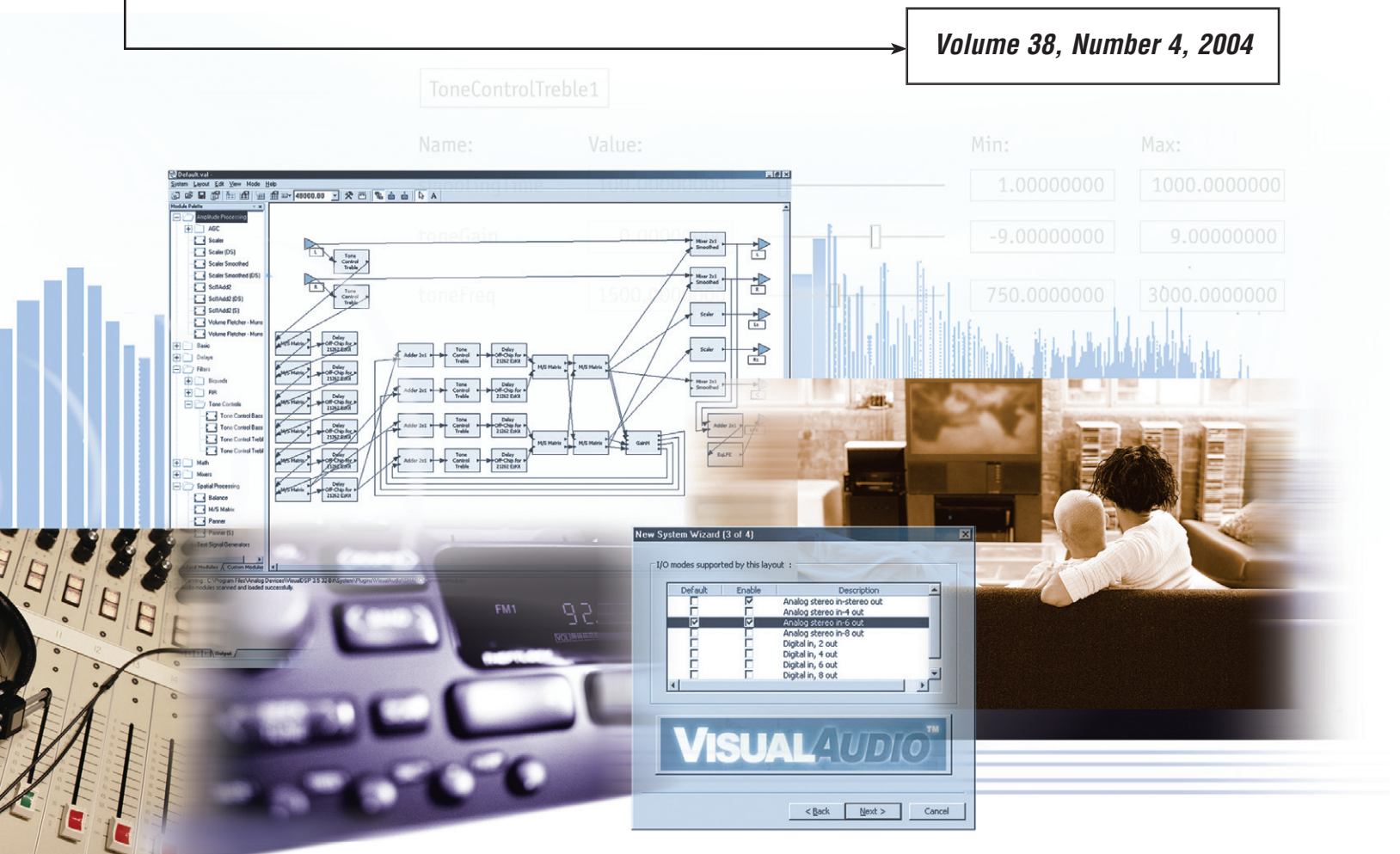


Analog Dialogue

A forum for the exchange of circuits, systems, and software for real-world signal processing

Volume 38, Number 4, 2004



In This Issue

Editors' Notes	2
Wideband CMOS Switches (Ask The Application Engineer—34)	3
Support for the Designer—Improved ADI Website Helps You	8
Designing Efficient, Real-Time Audio Systems with VisualAudio™	11
Recent Product Introductions	15
Authors	15

Editors' Notes

You've opened the book on the final quarterly issue of 2004, our 38th sequential year in print—and 6th online, at analog.com/analogdialogue. Perhaps you've read all four issues cover to cover. Or perhaps this is your first acquaintance with *Analog Dialogue*. In any event, here's your opportunity to spend a moment to be tempted to read an article you may have missed—or to contemplate a title that you've already read. You can find copies of all these issues online in the archives at <http://www.analog.com/library/analogdialogue/archives.html>



The year started—in Number 1—with PID control algorithms, fan-speed in temperature control, and video technology in automotive safety. In this column, you could have read a rambling historical discourse on (mostly—but not entirely— analog) multipliers.

In the following issue—Number 2—you (could have) read about current measurement in solenoids for automotive controls, bridge amplification with digitally programmed gain and offset, and practical techniques to avoid op-amp instability due to capacitive loading. There was also a description of techniques we use for in-package trimming of a low-cost CMOS amplifier with wide bandwidth, offsets less than 65 μV and drifts less than 7 $\mu\text{V}/^\circ\text{C}$.

The penultimate issue—Number 3—had an “Ask The Applications Engineer” (#33) feature on *direct digital synthesis* (DDS), plus articles on JPEG2000 image compression and a digitally adjustable cable equalizer. You also could have read about a reader's discovery—in a NASA vehicle, in equipment designed before he was born—of an ingenious but deceptively simple hot-wire anemometer. Its design principle was described in a (still interesting) article on measuring fluid flow with a self-balancing bridge; originally appearing in our Volume 5, in 1971; it was reprinted in this issue.

And in these pages today you can read about our designer-oriented updated website, a new software tool for memory-efficient, real-time audio designs, and an “Ask The Applications Engineer” (#34) on wideband CMOS switches.

Thus we close the book on Volume 38 and look forward eagerly to Volume 39, which will commemorate Analog Devices's 40th year of providing the electronic industry with innovative products, guidance, and ideas for analog- and digital real-world signal processing solutions.

Dan Sheingold [dan.sheingold@analog.com]

Analog Dialogue

www.analog.com/analogdialogue

dialogue.editor@analog.com

Analog Dialogue is the free technical magazine of Analog Devices, Inc., published continuously for 38 years—starting in 1967. It discusses products, applications, technology, and techniques for analog, digital, and mixed-signal processing. It is currently published in two editions—*online*, monthly at the above URL, and quarterly *in print*, as periodic retrospective collections of articles that have appeared online. In addition to technical articles, the online edition has timely announcements, linking to data sheets of newly released and pre-release products, and “Potpourri”—a universe of links to important and rapidly proliferating sources of relevant information and activity on the Analog Devices website and elsewhere. The *Analog Dialogue* site is, in effect, a “high-pass-filtered” point of entry to the www.analog.com site—the virtual world of *Analog Devices*. In addition to all its current information, the *Analog Dialogue* site has archives with all recent editions, starting from Volume 29, Number 2 (1995), plus three special anniversary issues, containing useful articles extracted from earlier editions, going all the way back to Volume 1, Number 1.

If you wish to subscribe to—or receive copies of—the print edition, please go to www.analog.com/analogdialogue and click on <subscribe>. Your comments are always welcome; please send messages to dialogue.editor@analog.com or to these individuals: Dan Sheingold, Editor [dan.sheingold@analog.com] or Scott Wayne, Managing Editor and Publisher [scott.wayne@analog.com].

SATELLITE RADIO, MP3s, AND STREAMING AUDIO

Back in the days of analog LPs on vinyl, I owned over 500 record albums. Then, as an early adopter of compact discs, I bought all of my new music on CDs, and even started to replace some of my records. Soon I abandoned the turntable altogether and gave all of my albums to my brother. Sadly, a flood ruined all the records, but I shed nary a tear, exulting in the luxury of the newer, smaller, virtually indestructible CDs.



Yet lately I've realized that I rarely buy CDs anymore—and when I listen to them it's almost always in the car. In my family room I usually listen to one of the dozens of commercial-free, CD-quality audio channels that are available over the digital cable. It makes available a much wider variety of music, and lets me view trivia, history, and other information on the song, album, and artist. When at the computer, I listen to streaming audio from one of several providers, using one of the available media players. The small annual payment for this service makes it possible to listen to high-quality audio from over 1,000 stations and lets me download my favorite songs for a nominal additional fee.

At the gym, at the beach, or in the backyard, I listen to my MP3 player. Where do the MP3s come from? Most were ripped from my CD collection, but the newest ones are all downloads. Why buy the whole CD when I need buy only my favorite songs—for a fraction of the cost—and eliminate the storage problem at the same time.

In the car, I listen mostly to the radio, but am constantly annoyed and frustrated by the large number of commercials, especially at drive time. Although I listen to CDs in the car, those jewel boxes take up too much space and are too hard to open safely while driving. CDs in sleeves are more space efficient and are easier to handle, but they're sometimes hard to identify without their covers. An FM modulator lets me listen to my MP3 player in the car, but the audio quality is not as good as a CD, and it's sometimes difficult to find an unused radio frequency in Boston's busy metropolitan market.

Thus, one option that now tops the priority list for my new car is satellite radio, either XM or Sirius. In the early days of cable, skeptics wondered why people would pay to watch TV when they could watch it for free. Today, many people feel the same way about radio, but I look forward to the day when I can give my CD collection to my brother and rely on streaming media wherever I go.

Why am I writing about this here? Because as I drive to work each day I can feel proud that Analog Devices offers amplifiers, converters, and processors that enable satellite receivers, set-top boxes, computer audio, and MP3 players to be small, flexible, power-efficient, and inexpensive—all the while providing high quality and functionality—plus the software that helps developers to quickly bring these products to market.

Your comments are welcome.

Scott Wayne [scott.wayne@analog.com]

Ask The Application Engineer—34

Wideband CMOS Switches

By Theresa Corrigan [theresa.corrigan@analog.com]

Q: What is a CMOS wideband switch?

A: CMOS wideband switches are designed primarily to meet the requirements of devices transmitting at ISM (*industrial, scientific, and medical*) band frequencies (900 MHz and up). The low insertion loss, high isolation between ports, low distortion, and low current consumption of these devices make them an excellent solution for many high frequency applications that require low power consumption and the ability to handle transmitted power up to 16 dBm. Examples of applications mentioned later in this article, include car radios, antenna switching, wireless metering, high speed filtering and data routing, home networking, power amplifiers, and PLL switching.

Q: How do these switches come to be so much faster than typical analog CMOS switches?

A: To improve their bandwidth, wideband switches use only N-channel MOSFETs in the signal path. An NMOS-only switch has a typical -3 -dB bandwidth of 400 MHz—almost twice the bandwidth performance of a standard switch with NMOS and PMOS FETs in parallel. This is a result of the smaller switch size and greatly reduced parasitic capacitance due to removal of the P-channel MOSFET. N-channel MOSFETs act essentially as voltage-controlled resistors. The switches operate as follows:

$$V_{gs} > V_t \rightarrow \text{Switch ON}$$

$$V_{gs} < V_t \rightarrow \text{Switch OFF}$$

Where V_{gs} is the gate-to-source voltage and V_t is defined as the *threshold voltage*—above which a conducting channel is formed between the source and drain terminals.

As the signal frequency increases to greater than several hundred megahertz, parasitic capacitances tend to dominate. Therefore, achieving high *isolation* in the switches' *off*-state and low *insertion loss* in the *on* state for wideband applications is quite a challenge for switch designers. The channel resistance of a switch must be limited to less than about 6 ohms to achieve a low-frequency insertion loss of less than 0.5 dB on a line with 50-ohm matched impedances at the source and load.

As a departure from the familiar switch topology, inserting a shunt path to ground for the *off*-throw—and its associated stray signal—allows the design of switches with increased off-isolation at high frequencies. The FETs have an *interlocking finger* layout that reduces the parasitic capacitance between the input (RFx) and the output (RFC), thereby increasing isolation at high frequencies and enhancing crosstalk rejection. For example, when MN1 is *on* to form the conducting path for RF1, MN2 is *off* and MN4 is *on*, shunting the parasitics at RF2 to ground, as shown in Figure 1.

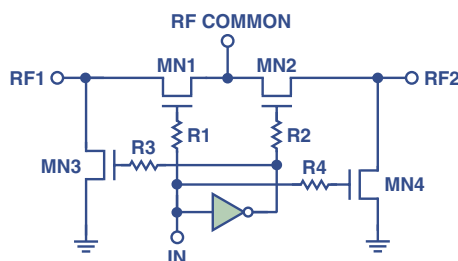


Figure 1. A typical transistor based Tx/Rx switch.

Q: You mention off isolation and insertion loss. Could you explain what these are?

A: Yes, the two most important parameters that describe the performance of an RF switch are the *insertion loss* in the closed state and the *isolation* in the open state.

Off isolation is defined as the attenuation between input and output ports of the switch when the switch is *off*. *Crosstalk* is a measure of the isolation from channel to channel.

For example, the ADG919 SPDT switch provides about 37 dB of isolation at 1 GHz, as shown in Figure 2. The same device, using the chip-scale package (CSP)—offered for space-constrained wireless applications, such as antenna switching—offers a 6-dB improvement (43 dB at 1 GHz).

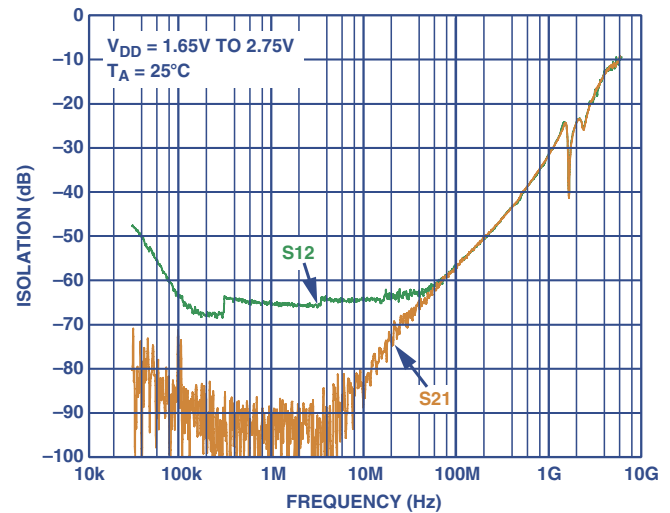


Figure 2. Off isolation vs. frequency.

Insertion loss is the attenuation between input and output ports of the switch when the switch is *on*. The switch is generally one of the first components encountered in a receiver's signal path, so a low insertion loss is required to ensure minimum signal loss. Low switch insertion loss is also important for systems that require a low overall noise figure.

To obtain the best insertion-loss performance from the ADG9xx family of switches, one should operate the part at the maximum allowable supply voltage of 2.75 V. The reason can be seen in Figure 3, which shows plots of insertion loss versus frequency for the ADG919 at three different values of supply voltage.

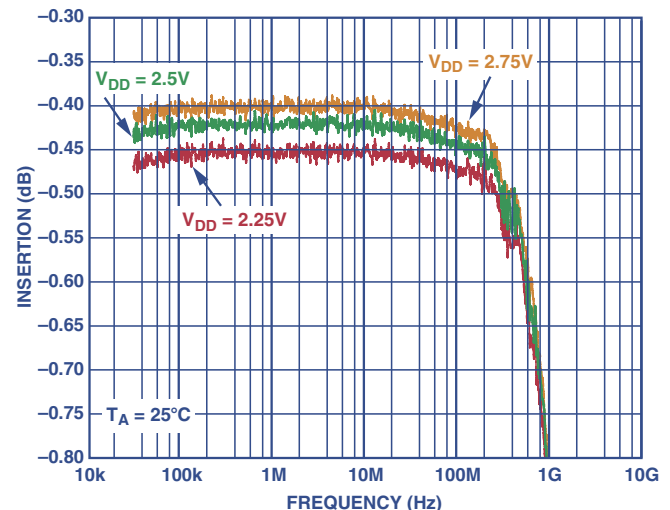


Figure 3. Insertion loss vs. frequency.

Q: How does insertion loss relate to the R_{on} -resistance spec of a standard analog switch?

A: Signal loss is essentially determined by the attenuation introduced by switch resistance in the *on* condition, R_{on} , in series with the source-plus-load resistance—measured at the lower frequencies of operation. Figure 4 shows a typical profile of *on*-resistance as a function of source voltage for an N-channel MOSFET device.

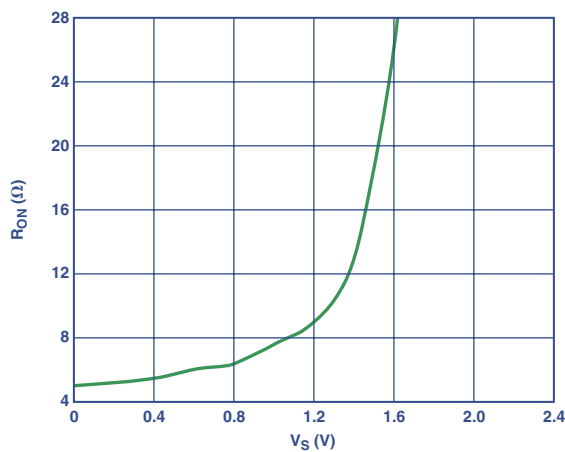


Figure 4. On resistance vs. source voltage.

Q: What technologies have been commonly used in the design of high-frequency switches?

A: Traditionally, only a few processes were available for developing good wideband/RF switches. Gallium arsenide (GaAs) FETs, PIN diodes, and electromechanical relays have dominated the market, but standard CMOS is now a strong entry.

PIN diodes are highly linear devices with good distortion characteristics, but they have many drawbacks given today's high performance demands. They have very slow switching times (microseconds, compared to nanoseconds for CMOS switches); they are power-hungry, making them unsuitable for many battery-operated devices; and—unlike CMOS switches with their response from RF to dc—there is a practical lower frequency limit to the use of PIN diodes as linear switches.

GaAs has been popular because of its low *on* resistance, low *off* capacitance, and high linearity at high frequencies. As CMOS process geometries continue to shrink, however, the performance of CMOS switches has increased to the extent that they can achieve -3 -dB frequencies of up to 4 GHz and are able to compete with GaAs switches. Designed to maximize bandwidth while maintaining high linearity and low power consumption, CMOS switches now offer a practical alternative to GaAs switches in many low-power applications.

Q: So what are the main benefits of CMOS wideband switch solutions over gallium arsenide?

A: Switches, such as the ADG9xx family of parts, have an integrated TTL driver that allows easy interfacing with other CMOS devices, since CMOS is compatible with LVTTTL logic levels. The small size of devices with integrated drivers is a solution for many space-constrained applications.

GaAs switches, as such, need dc-blocking capacitors in series with the RF ports, effectively floating the die relative to dc ground, so that the switches can be controlled with positive control voltages. Wideband switches, such as the ADG9xx family, do not have this requirement, eliminating concerns of reduced bandwidth, the impact of the capacitors on overall system performance, and the extra space and cost of GaAs solutions. Eliminating the blocking capacitors allows the ADG9xx parts to maintain their

low insertion loss (0.5 dB) all the way down to dc. In addition to providing a smaller, more efficient design solution, the ADG9xx family is less power-demanding, consuming less than 1 μ A over all voltage and temperature conditions.

Q: How about the ESD (electrostatic discharge) performance as compared to GaAs?

A: The ADG9xx family of parts passes the 1-kV ESD HBM (*human body model*) requirement. ESD protection circuitry is easily integrated on these CMOS devices to protect the RF and digital pins. This makes the switches ideal for any applications that are ESD sensitive, and they offer a reliable alternative to GaAs devices having ESD ratings as low as 200 V.

Q: What are the other important specifications of these switches?

A: **Video Feedthrough** (Figure 5) is the spurious dc transient present at the RF ports of the switch when the control voltage is switched from high- to low-, or low- to high, without an RF signal present. This is analogous to *charge injection* of a typical analog switch. It is measured in a 50-ohm test setup, with 1-ns (rise-time) pulses and a 500-MHz bandwidth.

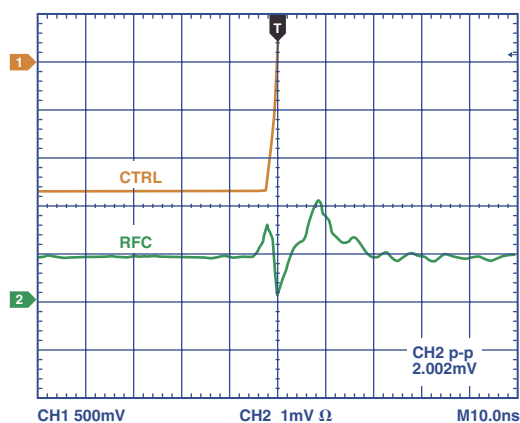


Figure 5. Video feedthrough.

P1dB (1-dB compression point) is the RF input power level at which the switch insertion loss increases by 1 dB over its low-level value. It is a measure of the RF power-handling capability of the switch. As shown in Figure 6, the ADG918 has a P1dB of 17 dBm at 1 GHz, with $V_{DD} = 2.5$ V.

Q: What does this mean?

A: It means that if the *insertion loss* at 1 GHz was 0.8 dB with a low-level input, it would be 1.8 dB with a 17-dBm input signal [Note: dBm is the dB (logarithmic) measure of the ratio of power to 1 mW, or voltage to 224 mV in 50 ohms. 17 dBm corresponds to 50 mW, or 1.6 V rms or 4.5 V p-p].

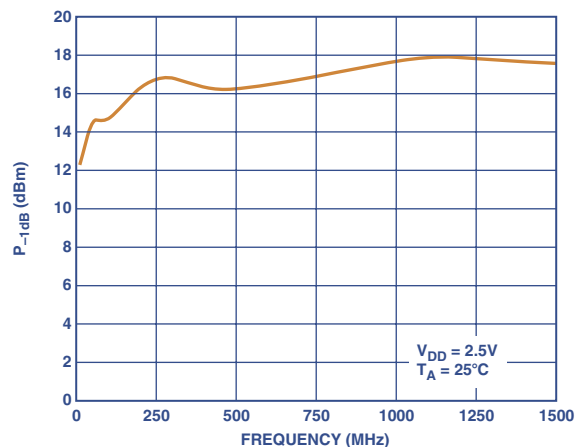


Figure 6. 1-dB compression point vs. frequency.

Q: Power-handling capability seems to decrease substantially at the lowest frequencies in Figure 6. Why?

A: In normal operation, the switches can handle a 7-dBm (5-mW) input signal. For a 50-ohm load, this corresponds to a 0.5-V rms signal, or 1.4 V peak-to-peak for sine waves. [$V_{p-p} = V_{rms} \times 2 \times \sqrt{2}$].

The power-handling capability is reduced at lower frequencies for two reasons:

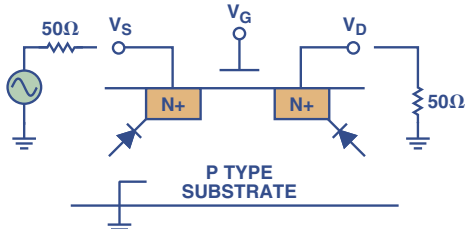


Figure 7. Physical NMOS structure.

As shown in Figure 7, the inherent NMOS structure consists of two regions of N-type material in a P-type substrate. Parasitic diodes are thus formed between the N and P regions. When an ac signal, biased at 0 V dc, is applied to the source of the transistor, and V_{gs} is large enough to turn the transistor on ($V_{gs} > V_t$), the parasitic diodes can be forward-biased for some portion of the negative half-cycle of the input waveform. This happens if the input sine wave goes below approximately -0.6 V, and the diode begins to turn on, thereby causing the input signal to be clipped (compressed), as shown in Figure 8. The plot shows a 100-MHz, 10-dBm input signal and the corresponding 100-MHz output signal. It is readily seen that the output signal has been truncated.

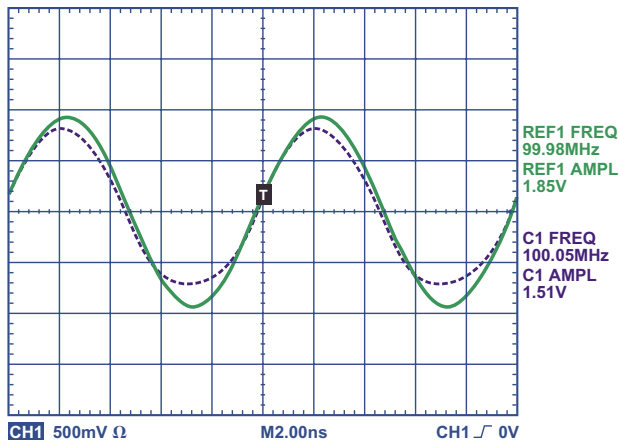


Figure 8. 100-MHz, 10-dBm input/output signals with 0-V dc bias.

At low frequencies, the input signal is below the -0.6 V level for longer periods of time, and this has a greater impact on the 1-dB compression point (P1dB).

The second reason why parts can handle less power at lower frequencies is the partial turn-on of the shunt NMOS device when it is supposed to be off. This is very similar to the mechanism described above where there was partial turn-on of the parasitic diode. In this case, the NMOS transistor is in the off state, with $V_{gs} < V_t$. With an ac signal on the source of the shunt device, there will be a time in the negative half-cycle of the waveform where $V_{gs} > V_t$, thereby partially turning on the shunt device. This will compress the input waveform by shunting some of its energy to ground.

Both of the above mechanisms can be overcome by applying a small dc bias (about 0.5 V) to the RF input signal when the switch is being used at low frequencies (<30 MHz) and high power—greater than 7 dBm (or 5 mW, 1.4 V p-p in 50 ohms). This will raise the minimum level of the sine-wave input signal and thus ensure that the parasitic diodes are continually reverse-biased and that the shunt transistor, never seeing $V_{gs} > V_t$, remains in the off state for the whole period of the input signal. Figure 9 again shows a plot of input- and output signals at 100 MHz and 10 dBm input power (about 2 V p-p in 50 ohms), but this time with a 0.5-V dc bias. It is clearly visible that clipping or compression no longer occurs at 100 MHz.

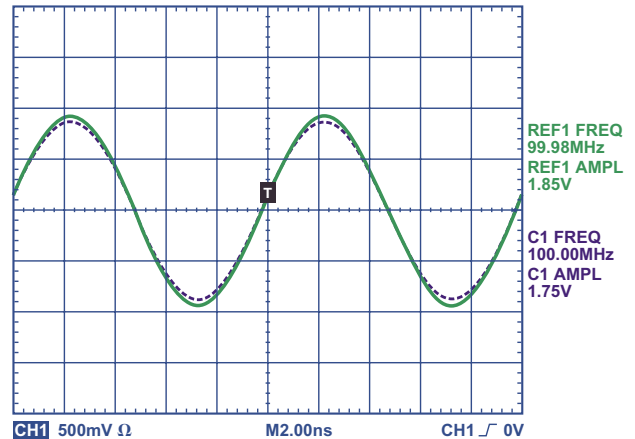


Figure 9. 100-MHz, 10-dBm input/output signals with 0.5-V dc bias.

Q: How do I apply a dc bias to RF inputs?

A: To minimize any current drain through the termination resistance on the input side, it is best to add the bias on the output (RFC) side. This is the best practice, especially for low-power portable applications, but it may be necessary to apply dc-blocking capacitors on the RF outputs if downstream circuitry cannot handle the dc bias.

Q: Can these switches operate with a negative supply?

A: They can operate with a negative signal on the GND (ground) pin as long as it adheres to the -0.5 V to $+4$ V Absolute Maximum Rating for V_{DD} to GND. Note that operating the part in this manner places the internal terminations at this new GND potential—an undesirable effect in some applications.

Q: What about the distortion performance of these switches?

A: When tones at closely spaced frequencies are passed through a switch, the nonlinearity of the switch causes false tones to be generated, causing undesired outputs at other frequencies. In communications systems, where channels are becoming more tightly spaced, it is essential to minimize this *intermodulation distortion* (IMD) to ensure minimum interference. Applying two closely spaced equal-power signals with a set frequency spacing (e.g., 900 MHz and 901 MHz) to the input of a device under test (DUT), results in the output spectrum shown in Figure 10. The 3rd-order harmonic, usually expressed in dBc, is the log of the ratio of the power in the 3rd order harmonic to the power of the fundamental. The larger the (negative) value, the lower the distortion. Sending these tones through the ADG918, using a combiner with an input power of 4 dBm, resulted in an IP3 of 35 dBm as shown in Figure 11. [Note: an excellent discussion of various types of distortion can be found in “Ask The Applications Engineer—13”]¹

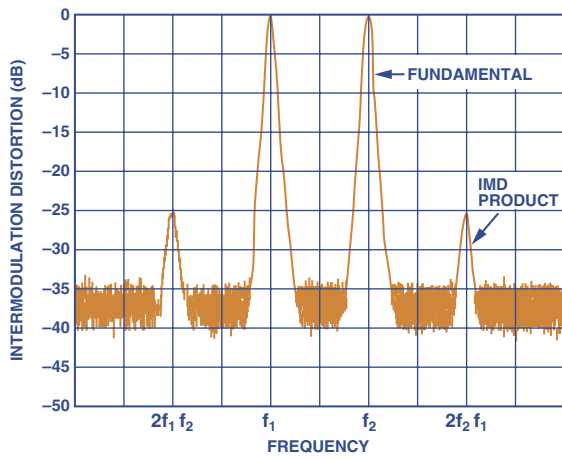


Figure 10. Output spectrum of two-tone IMD test.

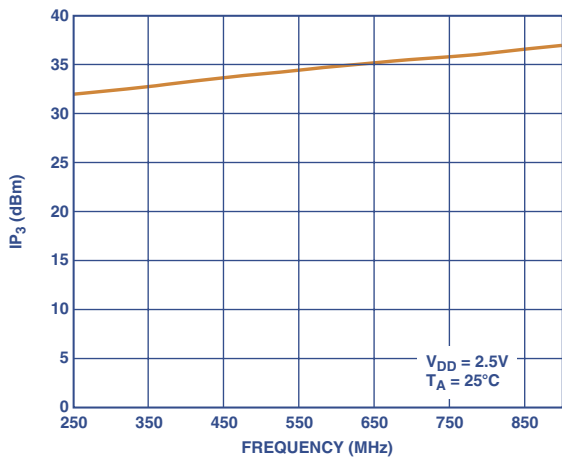


Figure 11. IP3 vs. frequency.

IP3—Third-order intercept point. The IMD is measured, and from this the IP3 value is calculated. IP3 is a figure of merit—in dBm—for the device. IP3, specified in the data sheet, is a measure of the distortion caused by the switch due to the power in these false tones. The larger the IP3 value the smaller the tones in the adjacent channels, indicating that the switch has good harmonic performance.

Q: What configurations are available in the ADG9xx family?

A: The ADG9xx family comprises SPST (*single-pole, single-throw*), SPDT (*single-pole, double-throw*), and dual-SPDT switches—and 4:1 single-pole *multiplexers* (SP4T). These are offered in both *absorptive* and *reflective* versions, in order to suit all application needs.

Q: What is an absorptive switch?

A: The ADG901 (SPST), ADG918 (SPDT), ADG936 (dual SPDT), and the ADG904 (SP4T) parts are described as *absorptive* (matched) switches, because they have on-chip 50-ohm-terminated shunt legs.

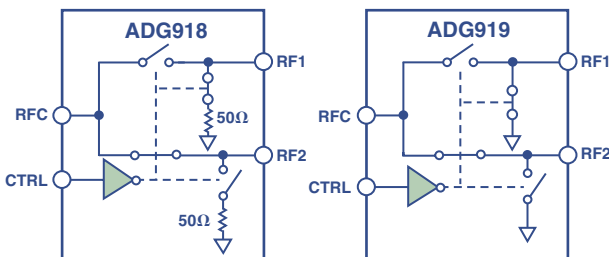


Figure 12. ADG918, an absorptive switch, and ADG919, a reflective switch.

Q: What is a reflective switch?

A: The ADG902 (SPST), ADG919 (SPDT), ADG936R (dual SPDT), and the ADG904R (SP4T) parts are described as reflective switches because they have 0-ohm shunts to ground.

Q: Where would I use an absorptive switch over a reflective switch?

A: An *absorptive* switch has a good impedance-match, or voltage standing-wave ratio (VSWR), on each port, regardless of the switch mode. It should be used when there is a need for proper back-termination in the *off* channel, to maintain a good VSWR. An absorptive switch is therefore ideal for applications that require minimum reflections back to the RF source. It also ensures that the maximum power is transferred to the load in a 50-ohm system.

A *reflective* switch is suitable for applications where high *off*-port VSWR does not matter and the switch has some other desired performance feature. Reflective switches are commonly used in applications where the matching is provided elsewhere in the system. In most cases, an absorptive switch can be used instead of a reflective switch, but not vice versa.

Q: How can I determine the VSWR of these switches?

A: **VSWR—voltage standing-wave ratio**—the ratio of the sum of forward and reflected voltages to the difference of forward and reflected voltages—indicates the degree of impedance match present at the switch RF port. When it comes to measurement, it is easier to describe the impedance match in terms of *return loss*, the amount of reflected power relative to the incident power at a port.²

Simply by measuring both incident and reflected power, the return loss can be determined, and from this the VSWR can be calculated by using readily available VSWR/return-loss conversion charts. Figure 13 shows a typical return-loss curve for the ADG918 in the *on*- and *off* conditions. Note that the ADG918, an *absorptive* switch, has good return-loss performance for the *off*, as well as the *on*, switch. The ADG919 version, which does not include termination resistors, would not have good return-loss performance in the *off* condition.

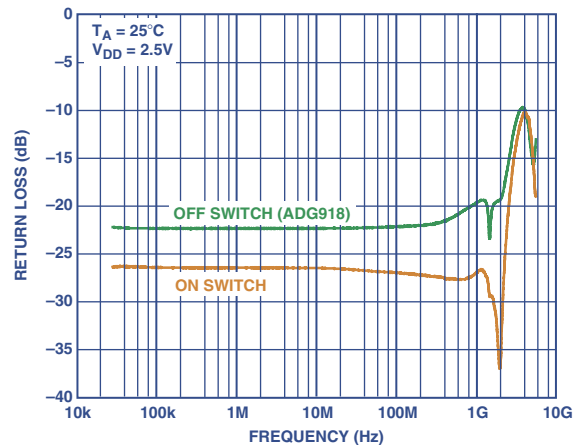


Figure 13. Return loss vs. frequency for the ADG918 switch.

Q: Now that you've explained how these parts perform, tell me where and how they are used.

A: Due to their low insertion loss at up to 1-GHz and wide -3-dB bandwidth (up to 4 GHz), switches in this family are ideal for many **automotive entertainment** systems.

They have found homes in **tuner modules** and **set-top boxes** to switch between the cable-TV input and the *off-air* antenna input. Another area where these parts are suitable is in **car-radio antenna switching**. Because these are

generally 50-ohm-impedance systems, the 50-ohm internal terminations offered by the *absorptive* version of these switches—the ADG901, ADG918, and ADG904—ensure excellent impedance-matching and minimum reflections.

The variety of topologies available makes these parts very easy to design into **antenna-diversity-switch** applications, allowing the user to switch between several antennas and a single tuner in multiband radios.

These parts are also suitable for **wireless metering** systems, providing the required isolation between *transmit* and *receive* signals (Figure 14).

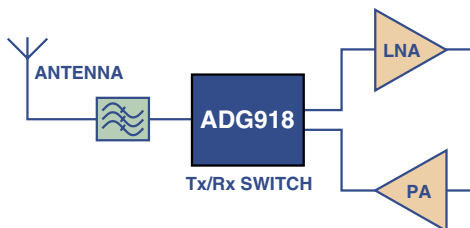


Figure 14. Tx/Rx switching.

These parts are perfect for **high-speed filter selection and data routing**: the ADG904 can be used as a 4:1 *demultiplexer* to switch high-frequency signals between different filters—and also to multiplex the signal to the output. For **differential filter selection** and data routing, the ADG936 dual SPDT (*single-pole, double-throw*) switch is an ideal solution. Data switching in modem cards for **point-to-point wireless systems**, such as microwave radio links for military and avionic applications, requires the high-frequency performance offered by the ADG9xx family of parts.

They are also suitable for **home-networking** applications—systems allowing the wireless remote control of many different functions, such as opening and closing roller blinds, control of lighting (*on, off* or dimming)—in which the information is transmitted through a wireless link. The excellent isolation performance at high frequency and low power-consumption preserve a system’s current budget—thus constituting an ideal application.

Due to their high frequency range—up to 4 GHz—this family of parts are also suitable for many **Bluetooth®** technologies—enabling wireless communication in the 2.5-GHz ISM frequency band.

Wideband switches can be used in the design of **power amplifiers** (PAs) with 800-, 900-, 1900-, 2100-MHz frequencies—for cellular CDMA and GSM applications. The switch is used in the feed-forward correction loop around the main amplifier, allowing the active- and passive feedback- and feed-forward paths to be switched out, permitting the amplifier’s distortion levels to be tested. The switch allows for gain- and phase correction in the system. The high isolation, low insertion loss, and the low distortion at 900 MHz make the ADG9xx family ideal for PA design in this frequency range.

The ADG918 can be used to implement **PLL switching** for frequency-hopping in GSM applications.

Q: What is PLL Switching, and why use the ADG918?

A: Switching between two *phase-locked loops* (PLLs)—commonly described as the *ping-pong* technique—allows a designer to achieve faster system settling times. The low power-consumption and simple single-pin control of the ADG918 make it an easy solution to integrate.

In switching between two oscillators, the desired isolation performance can be achieved by cascading—i.e., connecting a number of switches in cascade. This is a very simple way to provide a high-isolation specification for a system, preventing any interference at the higher frequencies. Cascading five ADG918s provides 130-dB isolation at 1 GHz, with an *insertion loss* of 3 dB. In this application, such an increase in insertion loss is not material, since the principal concern is about the signal levels relative to one another.

A nice feature of the ADG918 in this application is that it acts as an integrated **low-pass filter**, eliminating the unwanted harmonics created by the two PLLs. Achieved by the natural increase in insertion loss at high frequencies, it easily prevents the unwanted harmonics from propagating through the switches, as shown in Figures 15 and 16.

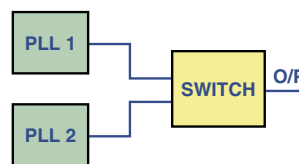


Figure 15. PLL switching application.

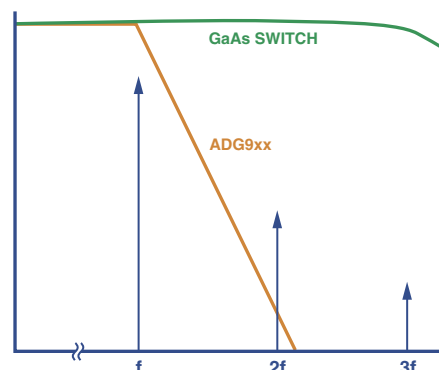



Figure 16. ADG918 switch cascade acting as integrated low-pass filter, compared with naked GaAs switching.

Q: So ... to summarize?

A: In summary, CMOS wideband switches, especially those in the ADG9xx family, are excellent choices for all applications in the ISM band that require high isolation and low insertion-loss for battery-operated devices with space constraints. Evaluation kits are available from Analog Devices to make the design-in of these parts fast and hassle free—every designer’s dream! 

NOTES

- ¹<http://www.analog.com/library/analogdialogue/Anniversary/13.html>
- ²<https://ewhdbks.mugu.navy.mil/VSWR.htm>

Support for the Designer— Improved ADI Website Helps You

By Pam Aparo [pamela.aparo@analog.com]
Reza Moghimi [reza.moghimi@analog.com]

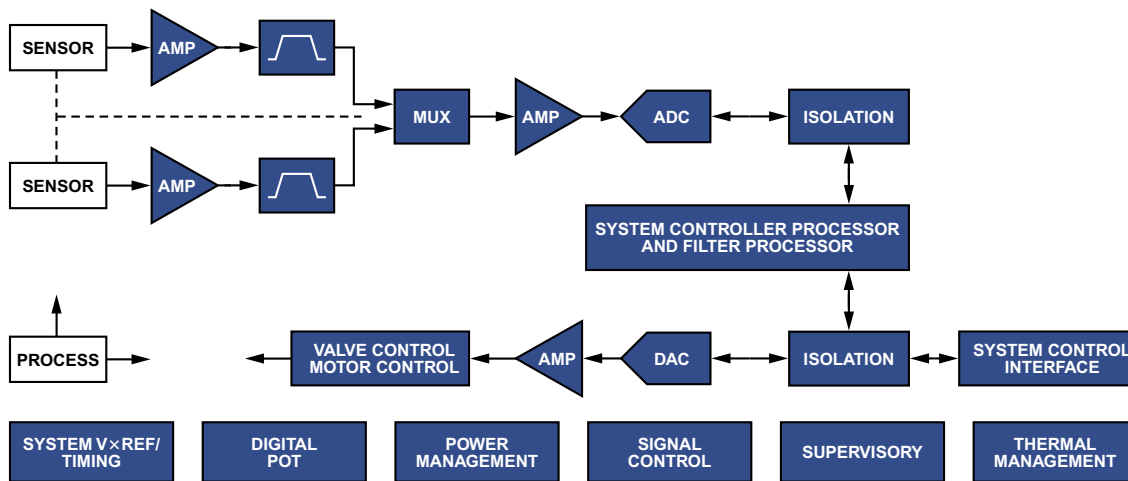
We have recently reconfigured the Analog Devices website to make it easier to use. If you are an old friend, perhaps you've already found that www.analog.com is friendlier than ever. If you are new to our website, Welcome! As you explore it, you'll find much useful information, and you'll have the ability to interactively find and apply the best products for your designs, using information listed under the two major tabs on the Home Page: *Products* and *Design Center*.

There you will discover utilities to find the right product for your application—and tools to help you use that product successfully. Utilities that assist in product selection include *parametric searches*, *selection tables*, and *signal chains*. Tools that assist in implementing the products include *models*, *interactive design tools*, *Analog Wizards*, *SimPLL*, and *SimADC*. These analog tools can be found in our Design Center (www.analog.com/designcenter), which also includes useful links to DSP tools, evaluation boards, and the library of technical information readily available from ADI.

The most powerful product selection tool on the site is the *parametric search*. When you know the specifications a device must have, the parametric search tool is the fastest way to find products to meet those requirements. The parametric searches allow you to specify a great many performance characteristics of a device. For example, the *op-amp* parametric search normally loads with 11 commonly used parameters—and 22 additional parameters can be added to the table to narrow the search. To keep the table from being too large, you can remove columns that are irrelevant to your design. The links to the analog-product parametric search tool can be found next to the *search* field on most pages of the corporate site.

Parametric searches are currently available for

- Op amps
- DACs
- Multiplexers
- Linear regulators
- References
- ADCs
- Switches
- Interface products
- Supervisory circuits



INDUSTRIAL MEASUREMENT/PROCESS-CONTROL SIGNAL CHAIN

Figure 1. A typical signal chain.

To use the parametric search, check the *include parameter* boxes to use the desired parameters, or uncheck the boxes to ignore those that are not of interest. Check the *priority* boxes to identify the parameters that are more important. To optimize a search on a specific parameter, type “best” in its *query parameter* field. For example—when looking for the lowest-power op amp that meets certain specs for headroom, supply voltage, and package—you would type “best” in the *quiescent current* column and specify the remaining requirements in the other columns. The search returns a list of all parts that meet the requirements. If there are fewer than 10 parts with the specs that are needed, the search will also return parts that almost meet your needs. Clicking on the down or up arrow for a given parameter will sort the list from highest to lowest or lowest to highest. Any required specs that are not met by a listed product will be highlighted in red. To get more information about a specific listed product, click on the model number to access its product page—which links to the data sheet.

Another product selection tool is the set of customizable *selection tables*, which can be found in the Design Center under Parametric Selection Tables, or under the Products tab. Selection tables are used to search through a smaller subset of products. Most of the selection tables on the Analog Devices site are *interactive*, allowing products to be sorted according to the parameters most important to the design. These tables let some parameters be selected with drop-down options; others can be sorted using the up/down arrows. For example, when using the interactive selection table for comparators, click on *customize table* for access to customizable parameters; select the logic level, then sort on the propagation delay. To get more information about a specific listed product, click on the model number to access the product page.

To find a part that ADI recommends for a specific application, a *signal chain* is a good place to start. Signal chains can be found at the top of the right-hand column of the Design Center. They are classified by market, and then by application within that market. The interactive signal chain for a selected application shows a block diagram implementation of that application (see Figure 1). The blue icons in the signal chain represent products that can be furnished by Analog Devices. Clicking on a blue icon will display a menu of product types that are applicable for that block. In the digital-camera signal chain, for example, clicking on audio amplifier will display the selections of audio amplifier, output amplifier, and volume control. Clicking on any of these will provide an interactive selection table of appropriate products. For each selected product, a single click will add it to a list that may be used for further in-depth study—or for ordering samples.

In addition to helping find the appropriate products, the Design Center has tools that can be used to shorten design time. Analog Devices has a suite of models, online tools, downloadable tools, and evaluation boards available. SPICE models are available for many op amps, instrumentation amps, references, and analog multipliers. ADI's SPICE models can be used to closely replicate transient and ac device performance. IBIS models, used to model the input and output characteristics of a device, are available for ADCs, DACs, and DSPs. *Saber* models are available for some instrumentation amps.

ADI also offers interactive design tools to predict the behavior of some devices. For many op amps and instrumentation amps, there are error-budget and voltage-range calculators. The voltage-range calculators are invaluable for working through an in-amp design in which the internal nodes can become saturated. Equations in the data sheet come to life when you can see the voltages on internal nodes attempt to exceed the supplies. Differential amplifiers are also more easily understood after using the online tool to see the effects of common-mode and differential voltage inputs.

Other interactive design tools are used to help configure a device for your application. For example, the ADF4110 has four 24-bit registers. The register maps are included in the data sheet, and have also been converted to an online tool. These register configuration tools can be used to double-check your own calculations, saving valuable design time. The website has *register configuration assistants* for many of our ADCs, PLLs, and DDS products.

Free downloadable tools are also provided to simulate the performance of PLLs and ADCs. The PLL simulation tool assists in selecting a PLL and VCO, designing a loop filter, and predicting the system's performance. The ADC simulation is a virtual evaluation board. By applying a behavioral model of the ADC to a collection of characterization data, an engineer can quickly decide which ADC will have the best performance with a specific input signal.

In addition to the interactive design tools that are specific to individual products, we offer several useful utilities. These include *power dissipation*, to predict how hot a device will become, *conversion* between dBc and dBm, and a calculator showing the relationship between SNR, THD, and ENOB.

Wizards are the colloquial name for *analog design assistants*, tools that offer a more in-depth treatment of an application. Currently, wizards are available for designing active filters and photodiode circuits—and interfacing with bridge circuits. They are discussed in some detail below. Other links at the design center include sample code, a list of evaluation boards, and the DSP knowledgebase.

24 × 7 Amplifier Applications Support: Analog Wizards

“How do I find the best amplifier to use for a photodiode application?”

“What is the best amplifier to use in a bridge-type application?”

“What is the best amplifier to drive my ADC's input circuit?”

These are some of the typical questions an application engineer hears everyday. The Analog Wizard is a powerful tool that helps *you* answer questions of this kind.

This web-based tool—a collection of *analog design assistants*—offers circuit designs based on a user's specific requirements and includes a list of the suggested components. In the near future, extensions to this collection will enable the Wizard to walk a user through the elements of a complete signal chain. These web-based tools can be bookmarked to be the starting destination for ADI website visitors pursuing designs using our products.

The Analog Wizard is a timely response to the inexorably increasing need for design assistance. The shrinking number of analog experts in the world of systems design; the difficulty of finding a system designer who can understand all the parameters of analog and digital ICs; and the vast number of similar-but-different offerings within

the product families of IC manufacturers have combined to create a need for this tool. This situation is further compounded by ever-shrinking product-design cycles, calling for ever-faster solutions.

Analog Devices, always aware of the needs for design assistance, has from the get-go helped designers by offering seminars, extensive publications—such as technical handbooks and in-depth data sheets—expert applications engineers, and—more recently—webcasts and web tools. Now, tools such as the Analog Wizard enable us to be at your beck and call, providing a high degree of applications support around the clock.

At first, the goal of the Analog Wizard was simply to narrow the choice to a few possible candidate amplifiers or other ICs to perform a specific function. But experience has shown that our role could not be limited to simply recommending suitable ICs; many users also requested that we specify the exact circuit configuration for their specific requirement. In one instance, we had to recommend an amplifier for a filter function, as well as to show the user how to implement the desired 3rd-order filter with a set of active and passive components. Thus, we needed to design and offer the complete circuit solution to the user. The Analog Wizard is designed to simplify the design task, providing many benefits to the user as discussed below.

The Wizard's operating approach is this: It asks the user a series of application-specific questions. Albeit not needing to know about the parameters of an amplifier or an ADC or microcontroller, the user *does* need to be able to specify exactly what is required. For example, the system designer needs to know the resolution requirements and available power supply levels in the system (whether single or dual, and their values). If at any point the user needs to understand the reasoning behind the questions asked, or is unclear about some of the terms used, the Wizard can help by providing links to definitions, articles, references, and examples of manufacturers' data sheets for explanations of the parameters and terms. The user can change the suggested default response to a question in order to see how it affects the solution.

Figure 2 shows an example of the types of questions asked about photovoltaic mode of a photodiode.

Parameter Name	Default Value	Your Value
1. Supply Voltage for Your System: (Range: 1.8 V to ±18 V)	±5 V	single supply ⊕+ <input type="text"/> V dual supply ⊖± <input type="text"/>
2. Photodiode's Capacitance: (Range: 15 pF to 1500 pF)	100 pF	<input type="text"/> pF
3. Photodiode's Output Impedance: (Range: 1 MΩ to 1 GΩ)	200 MΩ	<input type="text"/> MΩ
4. Photodiode's Responsivity: (Range: 0.1 A/W to 5 A/W)	0.5 A/W	<input type="text"/> A/W
5. Minimum Light Intensity: (Range: 400 pW to 400 nW)	4 nW	<input type="text"/> nW
6. Maximum Light Intensity: (Range: 401 nW to 4 mW)	100 μW	<input type="text"/> μW
7. Desired Bandwidth (BW): (Range: 100 Hz to 100 kHz)	10 kHz	<input type="text"/> kHz
8. Desired Full Scale Output: (Range: 1 V to 10 V)	5 V	<input type="text"/> V
9. Desired Accuracy: (Range: 8 bits to 16 bits)	12 bits	<input type="text"/> 15 bits

Figure 2. Typical Analog Wizard questions.

Based on the user's answers, the Analog Wizard calculates and defines its own search criteria for the Analog Devices *product database*. It then enters these search criteria into the product database, interrogates different part types that are suitable and relevant to the application, and suggests a number of them. The suggested parts are sorted based on required precision, temperature range, package options, and pricing. This listing and ranking might be presented in different ways, depending on the application. If the Wizard cannot meet all the imposed search criteria, it will still provide a close suggestion—with the failed parameters highlighted in red—and will state a good reason for the failure. Figure 3 shows a typical amplifier listing called up by a specific task.

In addition, the wizard allows the user to enter and compare a generic part number (for a part that may already be on hand) against the suggested parts. Further, the user can link to the product page of the selected device from the solution page.

Amplifier Part	1K Price (OEM US\$)	Available Packages	Temperature Range	Signal to Noise Ratio (Calculated) [dB]	Signal to Noise Ratio (Theoretical) [dB]	View Amplifier Solution
1. AD8571 BEST FIT	\$1.00	SOIC, SOP	-40 to +125 Deg C	93.24	92.06	View Amplifier Solution
2. AD8065	\$1.59	SOIC, SOT	-40 to +85 Deg C	90.60	92.06	View Amplifier Solution
3. AD8531	\$0.27	SC70, SOIC, SOT	-40 to +85 Deg C	94.83	92.06	View Amplifier Solution
4. AD8033	\$1.19	SC70, SOIC	-40 to +85 Deg C	89.26	92.06	View Amplifier Solution
5. AD8601	\$0.29	SOT	-40 to +125 Deg C	94.90	92.06	View Amplifier Solution

To compare a specific ADI part with the recommended parts, please type in the ADI part number and click the "Add to Table" button.

Figure 3. Suggested amplifiers for a photodiode application.

Besides the suggested ICs (usually up to 5 different device types), the user also receives a *bill of material* (BOM) and a circuit schematic representing the designed solution, as shown in Figure 4.

Amplifier Solution using the AD8571

Circuit schematic using the AD8571: [Perform Error Analysis](#) [Graph Noise Analysis](#)

SPICE Netlist(s)

- AD8571: SPICE Macro-Model Ver. 1.0, 10/99

Legal Disclaimer: Information furnished by Analog Devices is believed to be accurate and reliable. However, no responsibility is assumed by Analog Devices for its use, nor for any infringements of patents or other rights of third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of Analog Devices. Without limiting the foregoing, information from the Wizard is provided on an "as is" basis. Responsibility lies solely with the customer for any corresponding use of this information.

Bill of Materials - AD8571

1K Price (OEMUS):	\$1.00	Signal to Noise Ratio (Calculated):	93.24 dB
Available Packages:	SOIC, SOP	Signal to Noise Ratio (Theoretical):	92.06 dB
Temperature Range:	-40 to 125 Deg C	Supply Voltage (Vcc):	5 V
Feedback Resistance (Rf):	0.0900 MOhms		
Feedback Capacitance (Cf):	179.8388 pF		

Figure 4. Typical solution circuit with bill of material.

The Analog Wizard provides many more benefits. It will provide the PSpice model for ADI's IC device when available. It will also generate the circuit solution's *netlist*. Users can save or cut-and-paste this information into their PSpice model environment for further analysis and evaluation in a theoretical world.

The BOM shows the theoretical values of the passive components used in the circuit. Finding components in the real world that match the theoretical values will be very costly and time consuming, if not impossible, so the Analog Wizard allows the user to access the *error analysis tool*. In this environment, the user can introduce tolerances for resistance and capacitance values as shown in the schematic of Figure 5 and perform further error analysis on the suggested configuration.

Positive Supply Voltage for the Amplifier: 5

Negative Supply Voltage for the Amplifier: 0

I_{PD} : 5e-5 A

Feedback Resistance R_f : 0.09 MOhms

Ideal Opamp

Reset

$V_{OUT} = 4.5 V$

$V_{PD} = -1.109e-6 V$

Figure 5. Error analysis tool.

The maximum benefits from the wizard depend on the application at hand. For instance, in the case of the Photodiode Wizard, the user can look at the noise spectrum of the solution suggested. In addition, it provides the theoretical and actual *signal-to-noise ratio* (SNR). Figure 6 shows a typical noise spectrum for the circuit shown in Figure 4.

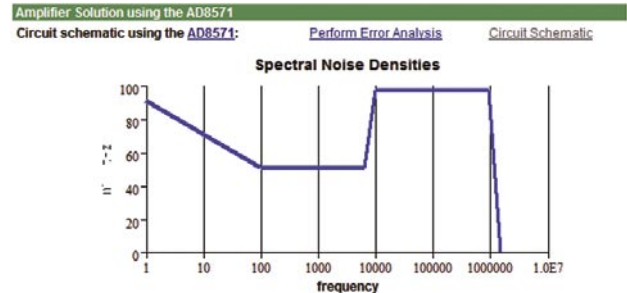


Figure 6. Typical noise spectrum for a photodiode application.

To simplify the task of getting user feedback, acquiring data on customer satisfaction, and providing any needed support, there are links for feedback and to request customer support on all pages of the Wizard.

The photovoltaic mode of photodiode application was picked for development due to its popularity and the number of questions asked by users. This could be followed in due course by development of a wizard to handle *photoconductive* circuit applications. The second module of Analog Wizard was the Bridge Wizard. The third module released was the Filter Wizard. The future of Analog Wizards looks very promising. There are many other persistently recurring topics, such as A/D converter drive amplifiers, currently handled by our application engineers that could be dealt with in mutually beneficial fashion by these *design-assistant*, or "Wizard," modules. Just to contemplate a signal chain, such as that shown in Figure 1, is to get a feeling for the opportunities in product and system design.

Before we leave the subject, we must return to real-world considerations and emphasize that these design *assistants* are just *that*. They can speed up the process and relieve the design engineer or technician of the arduous work of routine calculations—but they will never relieve the designer of the burden of *engineering judgement*. The list of real-world considerations beyond those handled by these simple "Wizards" is impressive: For example, the ever-increasing environment of RF energy as a source of interference must be dealt with; considerations of layout, grounding, strays, and parasitics will always be with us; and questions relating to the ambient physical environment, as it affects—and is affected by—the design always require attention.

Nevertheless, these tireless design assistants exist on the website to provide applications support around the clock. They are a resource that provides many benefits to designers. They are self-guiding and helpful; they ask a series of questions in familiar application-oriented language (rather than the language of IC parameters); they are helpful in the part selection process. They can offer the elements of a complete solution, including, as shown, a bill of materials and Pspice netlist; and they provide links to further error analysis and evaluation tools and materials—and ultimately to human assistance. Yet users can explore the possibilities at their own convenience, with powerful help toward designing the best solution using off-the-shelf parts. ▶

Designing Efficient, Real-Time Audio Systems with VisualAudio™

By Paul Beckmann [paul.beckmann@analog.com]
Vincent Fung [vincent.fung@analog.com]

The VisualAudio design and development environment is a new software tool for designing and developing audio systems. Its real-time-architecture is especially well-suited to the challenges of audio product development. This article briefly introduces VisualAudio, and then describes its framework, audio modules, and application to audio product development.

Audio Product-Development Challenges

Today, audio system developers are faced with the increasing need to design complex audio systems—especially home *audio/video receivers* (AVRs) and automotive infotainment systems—both quickly and cost-effectively. Why?

- The number of discrete audio channels in normal use has grown from 2 to 4 to 5.1—and most recently—to 7.1.
- The number of distinct, and sometimes competing, multi-channel audio formats has been increasing rapidly—to include, among others, Dolby® Pro Logic®, Dolby Digital, DTS® 5.1, Dolby Digital Surround EX™, and DTS-ES®.
- Products must interface with digital networks, such as the *Media Oriented Systems Transport* (MOST®) bus, requiring network stacks, content encryption and decryption, and sample-rate conversion—all within the audio processor.
- Consumers have come to expect sophisticated post-processing features, such as spatialization, automatic equalization, and bass management—in both *top-of-the-line* and mainstream products.

To deal with these factors, developers are turning to *digital signal processors* (DSPs), because their programmability allows systems to be customized for specific market niches and applications. The SHARC® Processor family from Analog Devices (ADI) is particularly well-suited for this task, since it offers features such as large internal memory, floating-point precision, and high-performance computation units. The recently announced *third-generation* SHARC processors take this one step further by integrating additional features that are specifically introduced to facilitate audio product design. These features include hardware sample-rate converters, encryption and decryption, a sophisticated digital audio interface, and an on-chip ROM containing multiple audio decoders.

The historical challenge faced by DSP users has been the development of software that makes optimum use of processor clock cycles and efficient use of memory. The long-used and laborious approach of hand-coding audio signal processing algorithms in assembly language has become less and less viable. This is particularly true when a large portion of the required effort goes into creating standard “checklist” and “me-too” functions instead of focusing on differentiating the product with value-added features. A better approach to developing audio product software was required.

To fulfill this need, ADI has developed a graphical environment—VisualAudio—as an aid to designing and developing audio systems that use the SHARC processor family. VisualAudio provides audio system developers with most of the software building blocks—together with an intuitive, graphical interface, shown in Figure 1—for designing, developing, tuning, and testing audio systems.

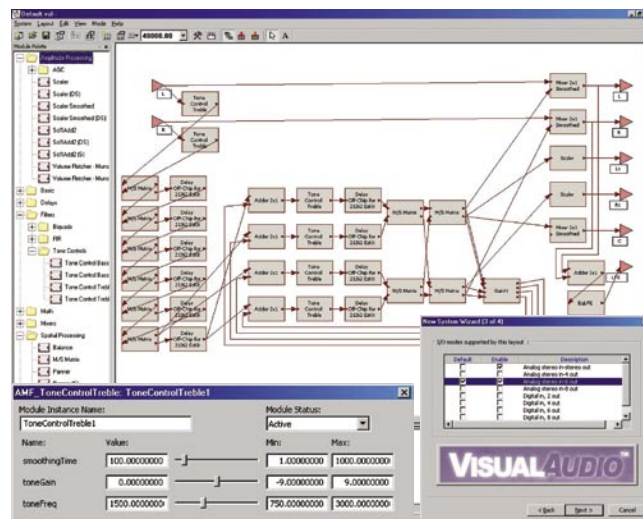


Figure 1. Example of VisualAudio graphical-interface screens.

VisualAudio comprises a PC-based *graphical user interface* (GUI, the graphical tool), a DSP *kernel* (the framework), and an extensible library of audio algorithms (audio modules). Working in conjunction with ADI's VisualDSP++™ *integrated development and debugging environment* (IDDE), VisualAudio generates product-ready code that is optimized for both *speed*, in millions of instructions per second (MIPS), and *memory usage*. By simplifying the process of developing complex digital signal-processing software, VisualAudio reduces development cost, risk, and time. As a result, audio-system developers are able to focus on adding value by differentiating their audio products from the competition.

At the core of VisualAudio is a real-time software architecture that handles audio I/O and post-processing. In order to be viable, the generated DSP code must be efficient in terms of MIPS and memory, and be flexible enough to handle a variety of audio product categories. The VisualAudio real-time architecture is described below, first the framework and then the audio-processing modules.

The Framework

The *framework* is the portion of the DSP code that handles system initialization, audio I/O, bit stream detection¹, instantiating and calling audio decoders, and communication with the host. VisualAudio provides its users with examples of frameworks for AVRs and automotive audio systems. By writing platform-specific drivers, VisualAudio users can customize many aspects of the framework to address specific product requirements. In some cases, ADI will also make the framework source code available to VisualAudio users if internal changes are necessary for optimum performance.

Audio products have specific requirements that govern the design of the framework. Each audio product has two primary functions: (1) real-time *audio processing*, and (2) *control* of this processing. The time scales for these two functions are vastly different. The real-time processing (with all internal operations completed) must occur at the sampling rate, otherwise there will be unacceptable pops and clicks in the output audio. The control functionality can occur at a much slower rate, 10 Hz to 100 Hz, and still be acceptable. The bulk of the MIPS usage thus occurs within the real-time processing, while the bulk of the *software complexity* is within the control functions. In order to simplify product design and development, VisualAudio separates the real-time and control functions into separate threads. Efficiency is achieved by hand-optimized, real-time audio processing modules, while the complexity of control code is managed by allowing the developer to write it in C and run it in a separate thread.

¹The bit stream detector, used with digital inputs such as an S/PDIF interface, monitors the incoming data stream and distinguishes between uncompressed PCM audio and compressed audio.

Traditionally, two different approaches to audio processing have been taken. In *stream processing*, audio samples are processed one at a time as they arrive, while in *block processing* several audio samples are buffered and then processed as a group. Each method has distinct advantages and disadvantages. Stream processing is efficient in terms of data memory, since no buffering of audio data is required. The major limitation of stream processing is that the overhead of multiple function calls cannot be tolerated. This forces the audio processing code to be written in-line, usually in assembly language. Such code is difficult to modularize and maintain.

Block processing requires additional buffering memory for I/O and scratch memory. Typical block sizes are in the range of 32 to 256 samples. Since many samples are processed at once, the overhead of function calls is amortized over a large number of samples. This leads to MIPS-efficient implementation—at the expense of additional memory—but is preferred, since structured programming methodologies may be employed. Block processing is also a natural fit to audio decoders that generate blocks of audio. For example, both Dolby Digital and DTS decoders generate audio in 256-sample blocks.

Block processing, the approach used by VisualAudio, has several additional advantages. All audio I/O within VisualAudio is double-buffered and managed using *direct memory access* (DMA). The processor receives an interrupt once per block—not once per sample—resulting in much less interrupt overhead than with stream processing. Also, by utilizing the chained DMA capabilities of the SHARC processors, double buffering is managed by the DMA controller, significantly increasing the allowable latency when servicing an audio input/output (I/O) interrupt.

The VisualAudio framework delivers audio to the post-processing network in blocks. Certain restrictions are placed on the block

size. First, it must be an even number, due to the *single-instruction, multiple-data* (SIMD) behavior of some audio modules. Second, the minimum block size is eight samples—due to pipelining within some audio modules. Finally, in systems with an audio decoder, the post-processing block size must be a factor of the decoder block size. For example, with Dolby Digital, possible block sizes are 8, 16, 32, 64, 128, and 256 samples.

Audio I/O and buffering can be seen within the example of a VisualAudio automotive framework, shown in Figure 2. Audio arrives from either the MOST network or from A/D converters, and is separated into multiple streams. The primary entertainment stream is generated by the DVD player, and additional monaural streams are produced by the telematics system or by chimes. The DVD data first undergoes *digital-transmission copy-detection* (DTCP) decryption, and is then fed to the bit stream detector. The output of the bit stream detector is packed into blocks; when a complete frame of data is available, the audio decoder is executed. The DVD player generates its own sampling rate, which is distinct from the sampling rate used by the post-processing. Thus, the output of the audio decoder must pass through an *asynchronous sampling-rate converter*. This block converts all input data streams to a fixed output sampling rate. At this point, the audio post-processing is executed at a fixed block size of 32 samples. As a final step, the audio channels are fed to D/A converters or returned to the MOST network.

The automotive framework contains multiple audio decoders, only one of which is active at a time. To reduce decoder memory requirements, VisualAudio manages a decoder memory pool that is shared among all possible decoders. The output of the decoder is fed to the post-processing network; and this drives the D/A converters.

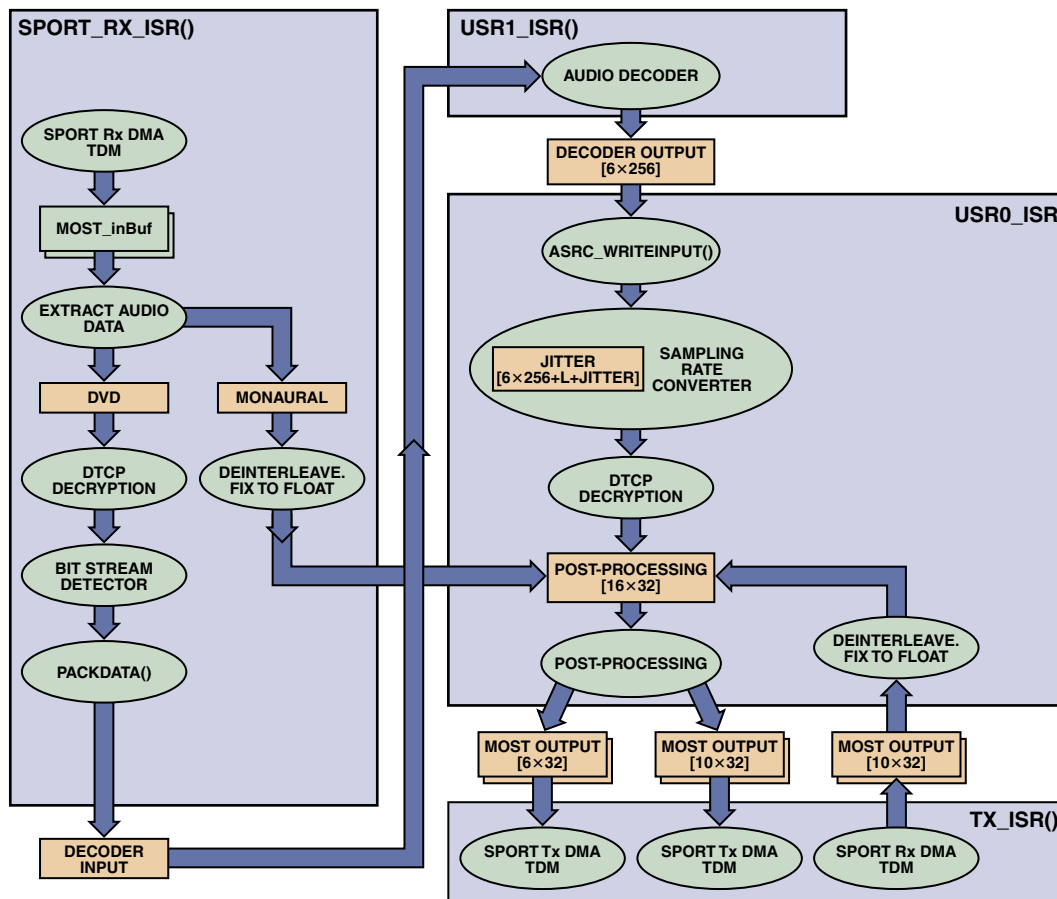


Figure 2. Audio flow within the VisualAudio automotive framework.

VisualAudio uses a simple interrupt driven kernel to manage multiple threads. For example, the sample automotive framework contains a total of six threads. From highest to lowest priority they are:

Host communication—exchanges messages with the host—typically via SPI. Messages are buffered and interpreted within the DSP *user control code* (described below).

Audio transmit interrupt—triggered by the serial port interrupt. Manages output DMA to the DACs—and formats data to be returned to the MOST network. Triggers audio processing within User Interrupt 0.

Audio receive interrupt—separates audio into distinct streams. Performs DTCP decryption and packs encoded data into frames. When appropriate, triggers the audio decoder within User Interrupt 1.

Audio processing (User Interrupt 0)—performs post-processing on a 32-sample block. The bulk of processing occurs within this thread.

Audio decoder (User Interrupt 1)—executes the audio decoding function.

DSP user control code (UCC)—runs whenever there are no interrupts active and serves as the *main loop* of the application. The user product’s control functionality occurs within this thread.

Typical thread activity within the sample AVR framework is shown in Figure 3. Each horizontal slice represents a different thread. The audio I/O, decoder, and the post-processing are run at regular intervals, and the UCC runs at the lowest priority. Note that the UCC might not be run for several milliseconds—while the processor is busy with audio processing.

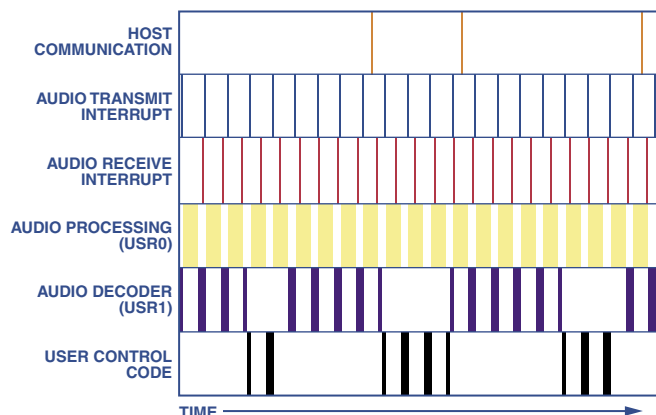


Figure 3. Thread activity within the VisualAudio AVR framework. The threads are ordered from high priority (at the top) to the lowest priority (at the bottom).

VisualAudio partitions control functionality between the host microcontroller and the DSP. This partitioning is arbitrary, and can even support systems without a dedicated host microcontroller. As described above, the UCC executes at lowest priority—using any free cycles not consumed by the interrupt handlers. The UCC is called periodically by the framework with *messages* and *notifications*.

Messages are commands sent from the host microcontroller to the DSP. These commands are used to control audio processing (e.g., “Set the volume to –20 dB,” “Set the bass tone-control to +3 dB”)

or to query the state of the audio processing (e.g., “Is the system limiting?”). The VisualAudio framework handles some commands internally, while the remainder are passed to the UCC. At each point in time, there can be only one pending message between the host and the DSP—and the DSP must send an acknowledgement after each command has been processed.

Notifications are generated asynchronously by the framework and occur under several conditions. The first notification occurs during system initialization—before any real-time processing is enabled. System- or application-specific initialization may be done at this time. A second notification, generated periodically at a rate of approximately 200 Hz, is used for control of the real-time audio processing, for example, *automatic gain-control* (AGC) calculations and updates. A final class of notifications is generated by the audio decoders in response to changes in the encoded bit stream. Notifications of this type occur when the sampling rate changes, the number of input channels changes, or if a *cyclic redundancy-check* (CRC) error is detected, in the incoming bit stream. These notifications allow the UCC to make appropriate changes in the audio processing.

Since the UCC is pre-empted by real-time audio processing, it may not be executed until several milliseconds have gone by—as shown in Figure 3. VisualAudio includes several features that simplify writing a UCC that will be constantly subjected to interrupts. First, since the host-communication interface only allows a single host message to be pending between the host and DSP, there is no danger of overflowing a message buffer, or of host messages overwriting each other. Another feature is a *notification queue*, in which notifications of the same type overwrite each other. For example, if two sampling-rate notifications are generated closely spaced in time before the UCC is executed, then the UCC will only receive the second notification—the final sampling rate. Also, since there are a finite number of notifications, the *notification queue* is necessarily of finite length.

The UCC must also be carefully written for updating certain audio module parameters. Some module parameters, such as *infinite impulse-response* (IIR) filter coefficients, must be updated automatically without being interrupted by the audio processing.

Each VisualAudio framework has an associated XML platform file that describes the capabilities of the target platform to the VisualAudio application, and also contains a list of the source, object, and library files needed to build the executable. Software design and development usually begins on a readily available evaluation or development platform, such as an EZ-KIT Lite evaluation kit from ADI, and then migrates to the actual target hardware once it is complete. VisualAudio’s Change Platform Wizard automates the process of migrating software between hardware platforms.

The Audio Modules

VisualAudio contains a library of approximately 100 audio-processing modules that have been optimized for the SHARC processors. The modules, categorized by function, include volume controls, tone controls, filters, mixers, etc.—enough functional types to develop a wide variety of audio products. These standard audio modules can be augmented by custom modules implementing proprietary post-processing features.

Figure 4 demonstrates the efficiency of block processing for a 10th order IIR filter—implemented as a cascade of five biquad filters. The number of operations per sample is plotted as a function of the block size. Most efficiency gains have been realized by a 64-sample block, with diminishing returns for larger blocks. For this filter, the core inner loop contains 21 multiply-accumulates (MACs) per sample; but by the use of SIMD instructions that operate on two pieces of data simultaneously, the loop is reduced to roughly 15 cycles.

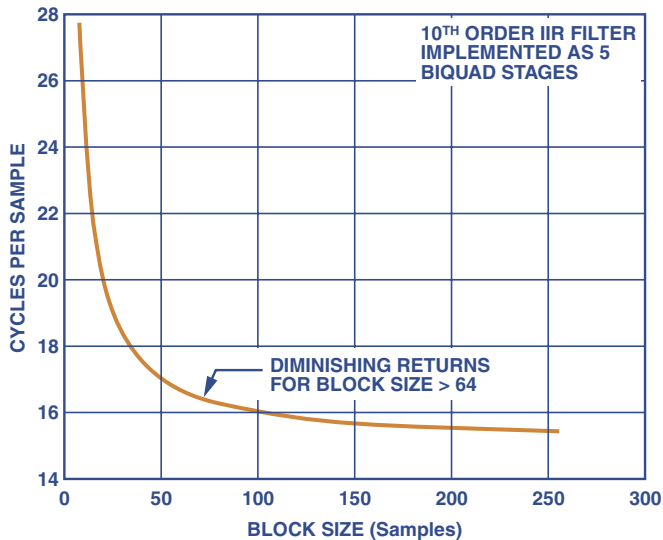


Figure 4. Processing efficiency in an IIR filter as a function of the block size.

Each audio module is described to the VisualAudio application by an associated XML file. This file contains a description of the module's in-memory data structure, memory allocation rules, input- and output audio channel list, high-level interface variables (shown on the module's graphical representation, or *inspector*), and design equations. The XML language consists of *tags* that label structured data. For example, consider a simple audio module that scales a monaural signal by a fixed gain. This module would contain a single *render Variable*—*amp*—that specifies the gain to apply. Within the module's XML file, the variable *amp* is described by the XML code shown below:

```
<renderVariable>
  <description type="string">amplitude applied
to input</description>
  <name type="string">amp</name>
  <float>
    <min type="float" modify="true">-1.0</min>
    <max type="float" modify="true">1.0</max>
    <default type="float">1.0</default>
  </float>
  <usage type="string">parameter</usage>
  <designSettable type="bool">true</
designSettable>
  <tunable type="bool">true</tunable>
</renderVariable>
```

The description tag provides a short summary of the variable. The name tag indicates the variable name within the data structure. The variable is described as being floating-point, having a default range of [-1 +1] that can be modified, and a default value of 1.0. VisualAudio will treat this variable as a design-settable

and tunable parameter, allowing it to be modified at design time, and later, during real-time tuning. This monaural fixed-gain example illustrates one of the many variable types that can be described by the VisualAudio XML format.

VisualAudio creates a separate data structure for each instance of an audio module within the post-processing layout. All data structures share a common 5-word header that describes the module's run-time interface to the framework. This is followed by module-specific parameters and state variables. For example, the type declaration of the data structure for the scaler module described above is:

```
typedef struct
{
  AMF_Module b;    // Common 5 word header
  float amp;      // Instance specific variable
} AMF_Scaler;
```

The audio processing functions within VisualAudio follow a uniform calling sequence. Each function is called with three arguments: a pointer to the instance data structure, a pointer to an array of input and output buffer pointers (input buffers first, then output buffers), and an integer specifying the block size. Continuing the example of the scaler, its real-time processing function is defined below.


```
void AMF_Scaler_Render(AMF_Scaler *instance,
float ** buffers,int blockSize) {
  int i;
  float *in = buffers[0];
  float *out = buffers[1];

  float amp = instance->amp;

  for (i=0; i<blockSize; i++) {
    out[i] = in[i] * amp;
  }
}
```

Note that this example is written in C to clarify its description. The actual *render* function included with VisualAudio is written in optimized assembly language for speed of execution.

SUMMARY

The complexity of digital signal processing software within audio products continues to increase. To address this, Analog Devices has developed VisualAudio, a graphical audio system design and development environment. VisualAudio simplifies audio product design and development by providing developers with many of the key software building blocks found in audio systems, managed with an intuitive graphical interface. The VisualAudio real-time architecture is flexible enough to support a variety of product types and is also efficient in terms of MIPS and memory. The overall VisualAudio tool-chain jump-starts product development; reduces development cost, time, and risk; and allows engineers to focus on innovation and product differentiation. 

More about VisualAudio

The VisualAudio design and development environment is expected to become available in November 2004, for use in conjunction with VisualDSP++ (release 3.5) for the SHARC Processor family. To take a VisualAudio test drive or learn more about VisualAudio and other Analog Devices development tools, visit: www.analog.com/processors/tools.

PRODUCT INTRODUCTIONS: VOLUME 38, NUMBER 4

Data sheets for all ADI products can be found by entering the model number in the Search Box at www.analog.com

October

Amplifier, Difference, single-supply, high common-mode voltage range .. AD8202
Audio Processor, 28-bit multichannel SigmaDSP® AD1940
Digital Downconverter, 150-Msps wideband AD6636
DAC, TxDAC+®, 16-bit, 200-/500-Msps AD9786
Gamma Reference, 12-channel, includes regulator and V_{COM} buffer ADD8707
Gamma Buffer, 18-channel, includes regulator ADD8709
Gyroscope, single chip with signal conditioning,
measures yaw rates up to 75°/s ADXRS401
Laser Diode Driver, dual-loop, operates from 50 Mbps to 3.3 Gbps .. ADN2870
Switch, CMOS, wideband, dual SPDT, provides
36-dB isolation at 1 GHz ADG936/ADG936-R

November

Accelerometer, ±5-g, small, thin package ADXL320
ADC, Successive-Approximation, 16-bit, 100-ksp/s AD7683
ADC, Successive-Approximation, 4-channel,
10-/12-bit, 188-ksp/s, I²C® interface AD7993/AD7994
ADC, Successive-Approximation, 8-channel,
10-/12-bit, 188-ksp/s, I²C interface AD7997/AD7998
Amplifier, Variable-Gain, quad, low-noise, low-cost AD8335
Amplifier, Operational, CMOS, dual, precision, low-noise,
low-distortion, 50-MHz AD8652
Controller, Synchronous Buck, 2-/3-/4-phase for Intel processors ADP3181
Controller, Synchronous Buck, 2-/3-/4-phase for AMD processors .. ADP3186
Frequency Synthesizer, PLL, includes voltage-controller oscillator .. ADF4360-8
Gamma Buffer, 18-channel, mask-programmable,
includes voltage regulator ADD8708
Line Driver, ADSL, upstream, rail-to-rail outputs AD45048
Touch-Screen Controller AD7877
Video Decoder, 9-bit, Multiformat SDTV ADV7181B
Video Decoder, 10-bit, Multiformat SDTV ADV7183B
Video Decoder, 12-bit, Multiformat SDTV ADV7189B

December

ADC, Successive-Approximation, 16-bit, 100-ksp/s,
true-differential inputs AD7684
ADC, Successive-Approximation, 16-bit, 250-ksp/s,
pseudo-differential inputs AD7694
ADCs, Sigma-Delta, 3-channel, 16-/24-bit, low-noise,
low-power AD7792/AD7793
ADC, Sigma-Delta, 6-channel, 24-bit, low-noise, low-power AD7794
ADCs, Successive-Approximation, 8-channel,
12-/10-bit, 1.5-Msps AD7938/AD7939
ADC, Successive-Approximation, 8-channel, 12-bit, 625-ksp/s AD7938-6
Amplifier, Difference, single-supply, high common-mode
voltage range AD8203
Amplifier, Instrumentation, precision *zero-drift*, rail-to-rail
inputs and outputs AD8230
Amplifier, Operational, dual, precision *zero-drift*,
rail-to-rail inputs and outputs AD8629
Amplifier, Operational, low-cost, high-speed, rail-to-rail output ... ADA4851-1
Battery Charger, Linear, 1.5-A for single-cell lithium-ion battery ADP2291
DAC, Voltage-Output, 40-channel, 14-bit, 3-V/5-V supply AD5384
DACs, Multiplying, dual 8-/10-/12-bit, wideband,
parallel interface AD5428/AD5440/AD5447
Digital Upconverter, multichannel, VersaCREST™
crest-reduction engine AD6633
Multiplexer, CMOS, quad 2:1, 300-MHz bandwidth,
HDTV audio/video applications ADG794
RF Splitter, Active ADA4302-4
Sequencers, Power Supply, supervise and monitor up to 10 supplies ... ADM106x
Supervisory Circuits, Microprocessor, include watchdog
and manual reset ADM63xx
Supervisory Circuits, Microprocessor, include watchdog
and manual reset ADM823/ADM824/ADM825
Switch/Multiplexer, CMOS, SDPT/2:1, 0.5-ohm *on*-resistance ADG839
Switch/Multiplexer, CMOS, dual 2-channel, 0.5-ohm *on*-resistance .. ADG884
Transceivers, RS-485/RS-422, half-duplex, low-power,
slew-rate-limited outputs ADM483
Transceiver, RS-485, half-duplex, isolated ADM2483
Transceivers, RS-485/RS-422, half- and full-duplex,
slew-rate-limited outputs ADM485x
Transmitter, FSK/GFSK/OOK/GOOK/ASK, multichannel,
ISM band ADF7012
Vector Multiplier, RF/IF ADL5390
Video Encoders, Multiformat, include six 12-bit
noise-shaped video DACs ADV7320/ADV7321

AUTHORS

Pam Aparo (page 8) received a BSEE degree from the University of Connecticut. After working in manufacturing and medical instrumentation, Pam joined ADI as an applications engineer and has contributed to ADI's website redesign. She recently transitioned to a new role as the Worldwide Distribution Training Manager. In her spare time, she enjoys tackling home improvement projects and training her dogs.



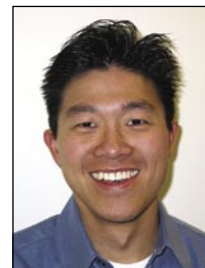
Paul Beckmann (page 11) is an independent consultant working with the Analog Devices Audio Rendering Technology Center (ARTC) in San Jose, CA. He specializes in audio system design and development, including algorithms, efficient implementations, and overall system architecture. Paul spent nine years at Bose Corporation in R&D and product development. Paul received SB, SM, and Ph.D. degrees in Electrical Engineering, all from MIT.



Theresa Corrigan (page 3), an applications engineer at ADI's facility in Limerick, Ireland, provides applications support for switch/multiplexer products and for general-purpose digital-to-analog converters (DACs). Theresa, who holds a BSc. (Hons) in Experimental Physics from National University of Ireland, Galway, joined Analog Devices in 2001. Her interests include playing soccer, reading, and traveling.



Vincent Fung (page 11) has been the Product Marketing Manager for Platform Tools Group's Audio Rendering Technology Center in Santa Clara, California, since 2002. Prior to joining Analog Devices, he has held key product-marketing and business-development positions at both established-high-technology companies and digital-audio distribution start-ups. He holds both Bachelor and Master degrees in Electrical Engineering, as well as an MBA in Marketing, from Columbia University.



Reza Moghimi (page 8), an ADI Applications Engineer in San Jose, received a BSEE from San Jose State University in 1984 and an MBA in 1990—and has also received a number of on-the-job certificates. He has worked for Raytheon Corporation, Siliconix, Inc., and Precision Monolithics, Inc. (PMI)—which was integrated with Analog Devices in 1989. At ADI, he has served in test-, product-, and project-engineering assignments. He has written many articles and design ideas—and has given presentations at technical seminars. His hobbies include travel, music, and soccer.



Analog Dialogue

Analog Devices, Inc. Worldwide Headquarters

Analog Devices, Inc.
One Technology Way
P.O. Box 9106
Norwood, MA
02062-9106 U.S.A.
Tel: 781.329.4700
(800.262.5643,
U.S.A. only)
Fax: 781.461.3113

Analog Devices, Inc. Europe Headquarters

Analog Devices SA
17-19 rue Georges Besse
Antony, 92160
France
Tel: 33.1.46.74.45.00
Fax: 33.1.46.74.45.01

Analog Devices, Inc. Japan Headquarters

Analog Devices, KK
New Pier Takeshiba
South Tower Building
1-16-1 Kaigan, Minato-ku,
Tokyo, 105-6891
Japan
Tel: 813.5402.8210
Fax: 813.5402.1064

Analog Devices, Inc. Southeast Asia Headquarters

Analog Devices
22/F One Corporate Avenue
222 Hu Bin Road
Shanghai, 200021
China
Tel: 86.21.5150.3000
Fax: 86.21.5150.3222



Purchase of licensed I²C components of Analog Devices or one of its sublicensed Associated Companies conveys a license for the purchaser under the Philips I²C Patent Rights to use these components in an I²C system, provided that the system conforms to the I²C Standard Specification as defined by Philips.

© 2005 Analog Devices, Inc. All rights reserved.
Trademarks and registered trademarks are the property of their respective owners.
Printed in the U.S.A. MO2000384-XX-2/05



www.analog.com/analogdialogue