TESTING: PWM Voltage Reference Technique
LOCATION: United States
PAGE: 18

DESIGN TIP: Protect Against EMI
LOCATION: Canada
PAGE: 46

Q & A: Embedded Theory & Practice
LOCATION: Israel
PAGE: 50

CIRCUIT CELLAR

THE WORLD'S SOURCE f OR EMBEDDED ELECTRONICS ENGINEERING INf ORMATION

NOVEMBER 2011
ISSUE 256

ANALOG TECHNIQUES

Analog Signal Management & Sound Tone Detection

MCU-Based Auditory Navigation System

Engineer an Alternative to Joystick Control

Electronics Design: Processors, Power, & Interfacing

A Look Inside an Ionization Detector

PLUS
"Spark" a Design
Implement 32-Bit Power & Arduino Compatibility

// Get Started with the chipKIT
// Program with a Modified IDE
// Shields Explained
// And More

7 25274 75349 9 1 1>
$7.50 U.S. ($8.50 Canada)

www.circuitcellar.com

TASK MANAGER

Tech Evolution

Like many of you, I have shelves of bookended *Circuit Cellar* magazines dating back to the first issue. For work-related purposes, and out of curiosity, I frequently take time out of my busy schedule to thumb through back issues to review many of the embedded designs of the past 23 years. As I prepared for this issue, I returned to *Circuit Cellar* 224 (March 2009) to reread Guido Ottaviani's article, "Robot Navigation and Control," which was actually the starting point for his article about sound tone detection in this issue. (When you get to his article on page 30, you'll understand why I returned to his 2009 robotics article.) After reading the article, I turned to the Task Manager section in that issue, where I had written:

> *Most quality new designs are the sum total of dozens, perhaps hundreds, of earlier projects, applications, and programs. When you see a project in this magazine, you can think of it as a single point in the long timeline of technological evolution. Looking ahead, you should consider each project, idea, and program described in these pages as a contribution to future projects.*

I still believe this. Guido's article in this issue is an excellent example. The project he finished in 2009 was simply a prelude to his work with sensors today. And, as you'll see, the other projects described in this issue are also contributions to the evolution of various important embedded technologies. Let's review.

On page 18, David Ludington continues his series titled "High-Accuracy Voltage Reference Using PWM." This month he covers a hardware design to evaluate practical PWM circuits.

Turn to page 24 for an article about an auditory navigation system project completed by a team of students at Cornell University. The team describes how it harnessed the power of various cutting-edge technologies—from embedded to GPS to biological—to build a wearable auditory navigator. The design is ready for future upgrades.

On page 42, Clemens Valens introduces the exciting chipKIT Max32. This is the next phase of Arduino-compatible technology.

Embedded system protection is a constantly evolving field of engineering. In this issue, George Novacek provides helpful information about EMI, EMC, and designing successful systems (p. 46).

Turn to page 50 for an interview with Shlomo Engelberg, a professor and author who has helped the field of embedded design evolve with his projects, courses, and books since the mid-1990s.

On page 54, George Martin presents the fourth article in his series on product development. His step-by-step tutorial will help you bring a product from idea to production.

On page 60, Richard Wotiz covers the topic of ionization detection. His article will likely inspire many of you to develop new, more effective detection systems of your own.

Lastly, Jeff Bachiochi presents the next phase of his innovative fly-by-wire wheelchair project (p. 66). This project represents not only the evolution of a novel system design, but also the evolution of more than two decades worth of electronics control technologies.

cj@circuitcellar.com

C. Abate

CIRCUIT CELLAR®

THE WORLD'S SOURCE FOR EMBEDDED ELECTRONICS ENGINEERING INFORMATION

Circuit Cellar's mission is to collect, select, and disseminate need-to-know information around the world in the fields of embedded hardware, embedded software, and computer applications. Circuit Cellar uses an assortment of print and electronic content-delivery platforms to reach a diverse international readership of professionals, academics, and electronics specialists who work with embedded, MCU-related technologies on a regular basis. Our aim is to help each reader become a well-rounded, multidisciplinary practitioner who can confidently bring innovative, cutting-edge engineering ideas to bear on any number of relevant tasks, problems, and technologies.

ADVERTISING

800.454.3741 • 978.281.7708 • www.circuitcellar.com/advertise

ADVERTISING REPRESENTATIVE
Peter Wostrel
Strategic Media Marketing, Inc.
1187 Washington St., Gloucester, MA 01930 USA
800.454.3741 • 978.281.7708
peter@smmarketing.us • www.smmarketing.us
Fax: 978.281.7706

ADVERTISING COORDINATOR
Kim Hopkins
E-mail: khopkins@circuitcellar.com

Cover photography by Chris Rakoczy—Rakoczy Photography
www.rakoczyphoto.com

PRINTED IN THE UNITED STATES

CONTACTS

SUBSCRIPTIONS

Information: www.cc-access.com, E-mail: subscribe@circuitcellar.com
Subscribe: 800.269.6301, www.cc-access.com, Circuit Cellar Subscriptions, P.O. Box 5650, Hanover, NH 03755-5650
Address Changes/Problems: E-mail: subscribe@circuitcellar.com

GENERAL INFORMATION

860.875.2199, Fax: 860.871.0411, E-mail: info@circuitcellar.com
Editorial Office: Editor, Circuit Cellar, 4 Park St., Vernon, CT 06066, E-mail: editor@circuitcellar.com
New Products: New Products, Circuit Cellar, 4 Park St., Vernon, CT 06066, E-mail: newproducts@circuitcellar.com

AUTHORIZED REPRINTS INFORMATION

860.875.2199, E-mail: reprints@circuitcellar.com

AUTHORS

Authors' e-mail addresses (when available) are included at the end of each article.

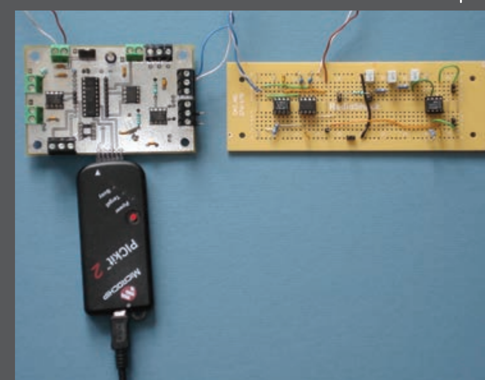
CIRCUIT CELLAR® (ISSN 1528-0608) is published monthly by Circuit Cellar Incorporated, 4 Park Street, Vernon, CT 06066. Periodical rates paid at Vernon, CT and additional offices. **One-year (12 issues) subscription rate USA and possessions $45, Canada/Mexico $60, all other countries $63. Two-year (24 issues) subscription rate USA and possessions $80, Canada/Mexico $110, all other countries $116.** All subscription orders payable in U.S. funds only via Visa, MasterCard, international postal money order, or check drawn on U.S. bank. **Direct subscription orders and subscription-related questions to Circuit Cellar Subscriptions, P.O. Box 5650, Hanover, NH 03755-5650 or call 800.269.6301.**

Postmaster: Send address changes to Circuit Cellar, Circulation Dept., P.O. Box 5650, Hanover, NH 03755-5650.

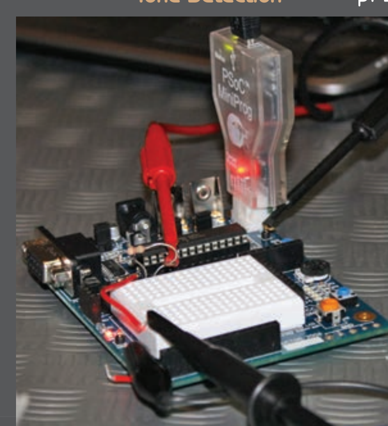
INSIDE ISSUE 256

November 2011 • Analog Techniques

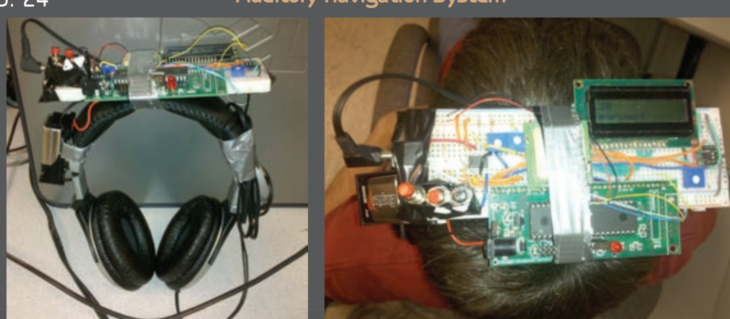
PWM Generator Board p. 18



Tone Detection p. 30



p. 24 Auditory Navigation System



by Guido Ottaviani (Italy)

Sound Tone Detection with a PSoC (Part 1)

Analog Signal Management

The first part of this two-part series details the hardware part of an audio sensor. You learn how to mount the component to a printed circuit board (PCB) and configure the internal hardware modules to manage an analog signal.

My Rino robotic platform is continuously evolving. Being a development platform it will, perhaps, never be completed (www.guiott.com/Rino/index.html). As shown in my "Robot Navigation and Control" article series (*Circuit Cellar* 224 and 225, 2009), the platform can autonomously navigate in an unknown environment. In "A Sensor System for Robotics Applications" (*Circuit Cellar* 236, 2010), I explained how to build a dedicated subsystem to discover an obstacle on a robot path and find some targets using popular sensors readily available on the market. Now I want to focus on a specific kind of sensor, the most complicated one for people who are used to designing digital circuits: the audio sensor. Its design requires a good knowledge about analog circuits. Fortunately, modern op-amp chips help a lot in this type of design, reducing the complexity and instability typical of such circuits. But, you still need a good background in both practical and theoretical analog issues to avoid unwanted results. Once more, evolving technology helps us.

Cypress Semiconductor developed a programmable system-on-chip (PSoC) that contains several analog and digital blocks that easily connect together (see Photo 1). There are many "bricks" representing different kinds of functions and, like a Lego system, you have to arrange them in order to have a circuit that transforms the analog signal according to your needs. You still need some knowledge about analog electronic circuits; you must know the theory behind them. But, if you don't have enough practice building these kinds of circuits, you can still tackle such a project. You can use point and click instead of a soldering iron, you don't need to test many discrete passive components on a perfboard. You can simply reprogram your PSoC until the goal is achieved.

The PSoC is quite different from the usual microcontroller, CPU, or FPGA. It flashes a program inside to perform some tasks but it's not just a microcontroller. It mixes different electronic functional blocks, but it's not an FPGA. Its unique feature is that it has the potential to modify an analog signal keeping it analog and the characteristics of the analog blocks can be modified on-the-fly by the program. You have the alternative to use a digital-signal controller to sample your analog signal and apply

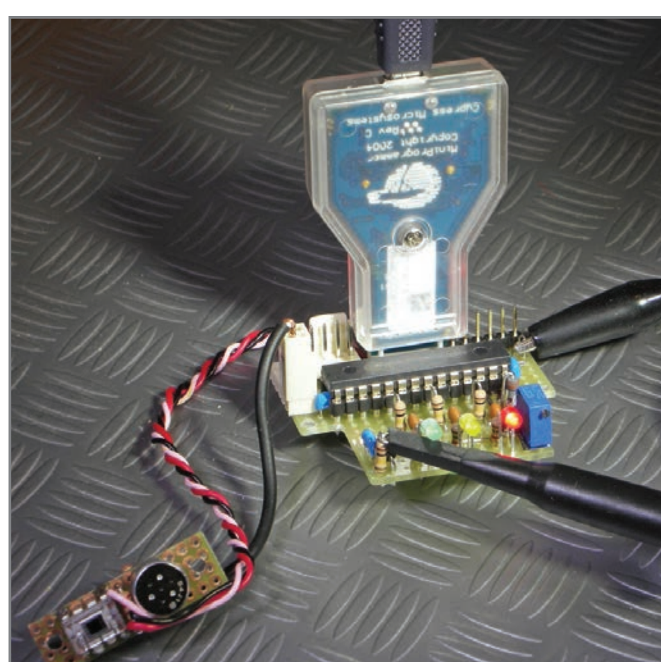


Photo 1—Cypress Semiconductor's PSoC and its programmer used in the real circuit

some mathematical transformations on it. But, you still need amplification, mixing, filtering, and amplification of the previous signal before sending it to an ADC for processing the signal by software. Often, a PSoC can do all this (and more) for you. The analog blocks are really analog! They are real operational amplifiers; they are not a software simulation of an analog behavior or a mathematical elaboration. The only digital stuff is that the capacitance can vary using switched capacitors technique. Adjusting the switching frequency and/or duty cycle, you can modify the capacitance, and therefore the parameters of the analog blocks, without adding external components. Looking at the electrical characteristics on the datasheets, you can find the same parameters of a normal op-amp datasheet: GBP, noise factor, common-mode rejection ratio, and so on.

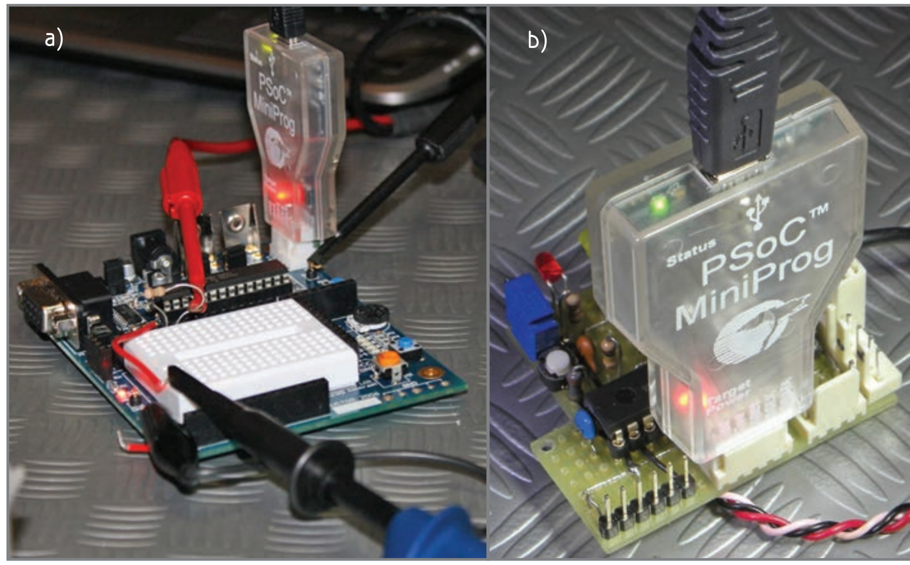


Photo 2a—You can buy a full evaluation kit with everything needed to start experimenting. b—Or you can buy the less expensive miniprogrammer to use with your own prototyping or final board. In both kits, there is a Cypress Semiconductor CY8C29466-24PXI PDIP PSoC device sample, which is perfect for our purposes.

As an example, there is a module that is unique on programmable devices—an instrumentation amplifier. This is one of the trickiest circuits to realize even with expensive op-amps. It requires good knowledge to obtain very high impedance, enough bandwidth, a good common-mode rejection ratio, and everything else needed to have a good measuring instrument. This is not achievable with a standard microcontroller.

I don't want to write a PSoC manual. I'm not going to explain every feature of PSoCs. I'll just explain how to build a tone-detector circuit with a Cypress device, focusing on some specific issues I've found. Even if you don't need a 4-kHz tone-detector circuit, the same basic principles I'm going to describe can be useful for many other types of projects. This could be considered "on-the-job training."

THE BASICS

Do you remember the circuit described in my sensor board article? In an explorer robotic challenge, the robot must autonomously find some sound sources that emit a 4-kHz tone. To achieve this goal, the robot has three microphones on three sides. The conditioning and revealing circuit is based on a quad op-amp and a tone detector. There is a first stage that mixes and amplifies the signals coming from the microphones, an active tunable band-pass filter, a second amplifier stage, and an NXP Semiconductors NE567 tone detector. The remaining op-amp in the chip is used as an active virtual ground reference for all the amplifiers, enabling higher impedance for their inputs.

At the very least, my first goal was to obtain the same functions with a single PSoC. I reached and exceeded that minimum goal, and I added some more functionality that I'll describe in this article.

First of all, what kind of PSoC? Let's start with the simplest and most affordable version, PSoC 1. This is an easy way to start understanding this device family without spending a fortune. The PSoC 1 series was the first series realized by Cypress and it's evolved noticeably both in analog module features and in digital block availability. There are many devices in this series, distinct for maximum clock frequency, RAM, and flash-memory quantity and for number of blocks. The latest one is the CY8C29x66 family—sold in different versions, from 24 to 64 pin—the only series available in a PDIP version, which is good for development boards (see Photo 2).

CONNECTING BLOCKS

Every book about PSoCs starts with, "Design and carefully test your circuit before you start writing the code." It's really true! When you build your project, after connecting all the blocks needed, Cypress Semiconductors's PSoC Designer integrated development environment (IDE) defines the template for the program you are going to write. It collects all the libraries needed and creates the basic main.c file and all the "include" files with the name you gave to each block. Changing the block configuration after you have written hundreds of lines of code using those names and libraries, even if possible, can be painful.

The good thing is that you can test your analog circuit without writing a line of code. Connect programmable amplifiers, filters, and other blocks via the common buses available and you can display the results of your work with a scope after injecting the signal to the input. It's not a simulation; it's real.

If you are used to working with microcontrollers, the first approach could seem strange. You don't have a fixed number

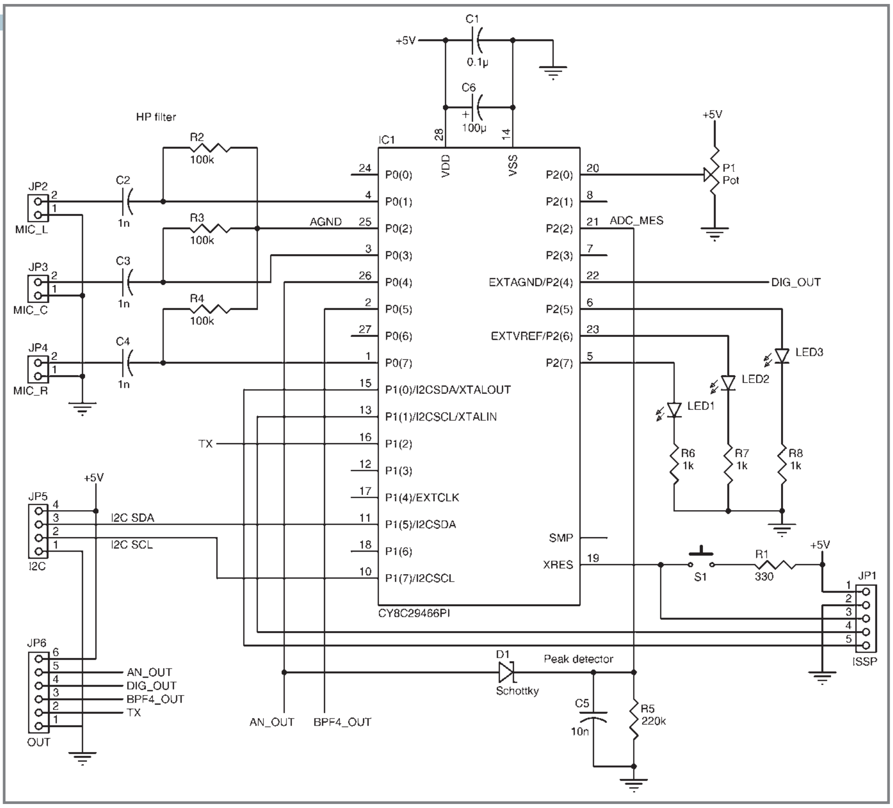


Figure 1—The "external" connections

even only as an exercise for the brain, to read some of the many application notes available on Cypress's website.

TRICKS & TIPS

Allow me to share some personal experiences with the hope of saving you some time and headaches. When copying text from one window to another, you cannot use standard ctrl-C and ctrl-V for copying and pasting. You must use drag-and-drop with a mouse between tabs. Fortunately, I'm a Mac OS X user, and drag-and-drop is more common than copy and paste for me, compared with Windows users. At least at the beginning, it is highly recommended to enable the "show allowed connections" feature (right click) to immediately know which connections are admitted for the line or module you are using.

Lastly, there is no big enough monitor when you are working with PSoC Designer, you have to zoom and pan often. To navigate more easily on the block diagram to place or connect modules and buses, you can temporary switch from standard to pan-mode view with an Alt key instead of clicking the menu item or using the right click contextual menu.

of peripherals that can be connected to I/O pins according to your needs; you have a fixed number of boxes that can be filled with any kind of functional block. In this case, the limit is the number of boxes, not the number of blocks. The blocks can be connected internally with other blocks or externally with I/O pins. Not every combination is possible. There are a good number of buses and boxes, but you have to study the right compromise between your needs and the connections available. It can be confusing at the very beginning, but it becomes easier after a while. Be prepared to spend most of your time testing different connections and different configurations for each of the blocks before reaching the optimum solution.

Every operation is performed inside PSoC Designer. You can connect blocks, configure them, read the datasheet of each block, write and compile your program, flash the device, and more without going outside of the IDE. You can also configure everything manually editing the code; after all, it acts as a usual microcontroller with a huge number of registers for a lot of different configurations. You have to take care of all the compatibilities avoiding possible conflicts. Fortunately, PSoC Designer does all these things, creating the code for you.

At the end, you can also create an exhaustive report of the entire configuration with the block diagram and a detailed description for each block using the menu item "view-configuration datasheet." It creates a folder (ConfigDataSheet) with all the information and the picture for a complete documentation. Many of the pictures used in this article have been automatically generated with this function. It is interesting,

THE CIRCUIT

The "external" circuit is simply a bunch of passive components (see Figure 1 and Photo 3). On the microphone's side, there are three high-pass RC filters for DC decoupling and low-frequency noise filtering. Three current-limiting

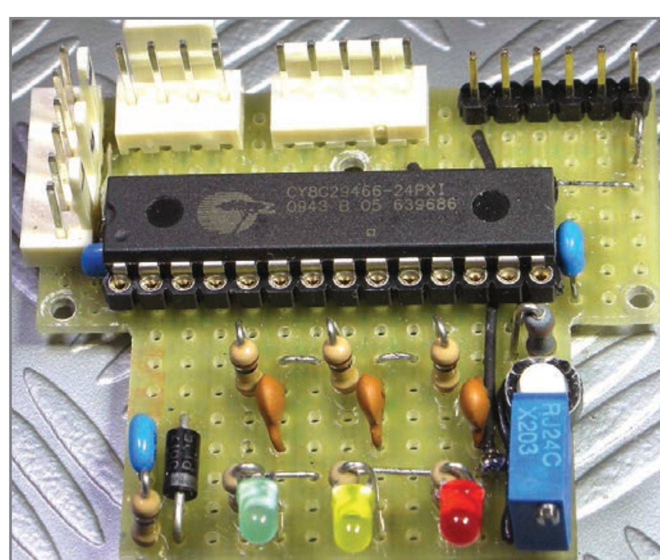


Photo 3—The circuit is simple. It is easy to wire it on a small piece of perfboard.

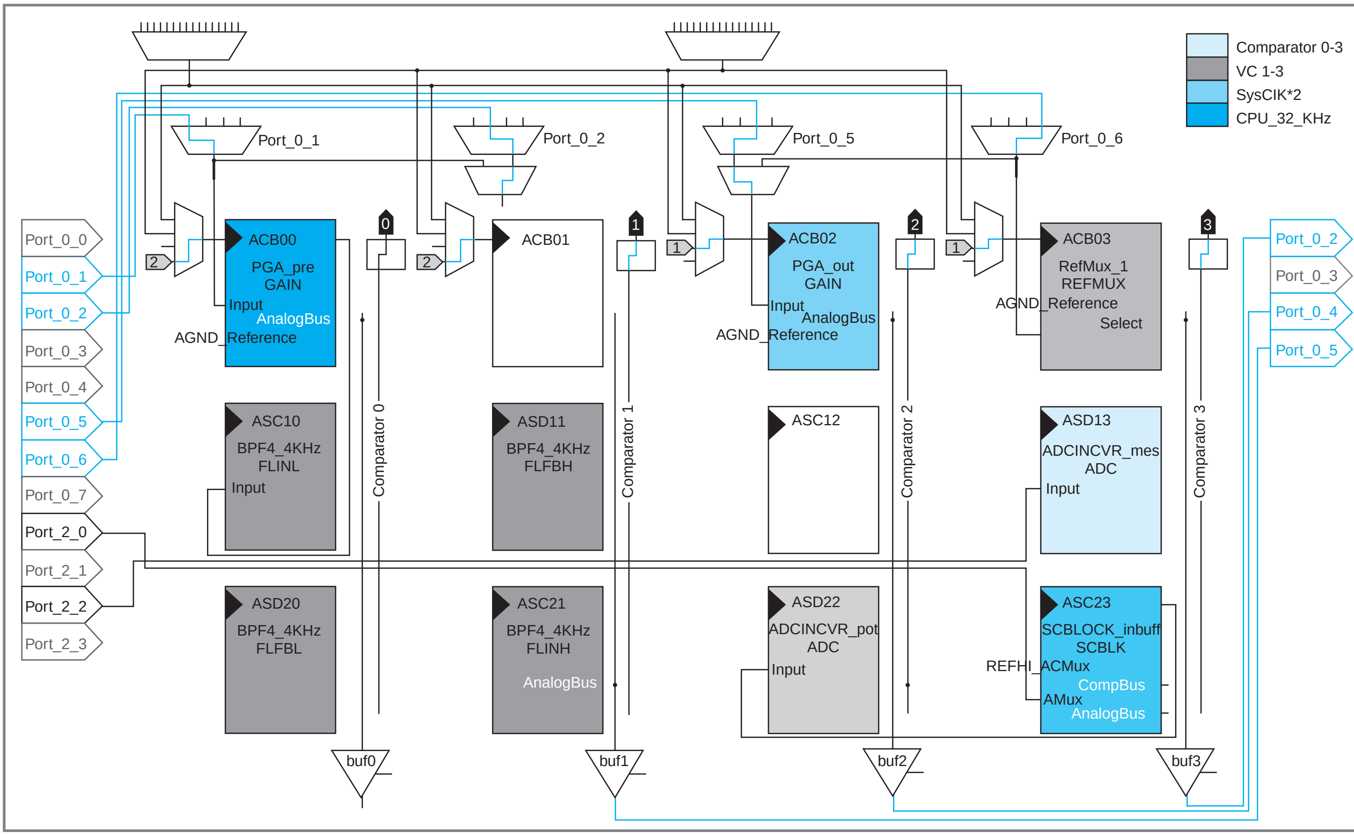


Figure 2—The analog blocks configuration with the connections to the internal buses and external pin drawn using PSoC Designer schematic conventions

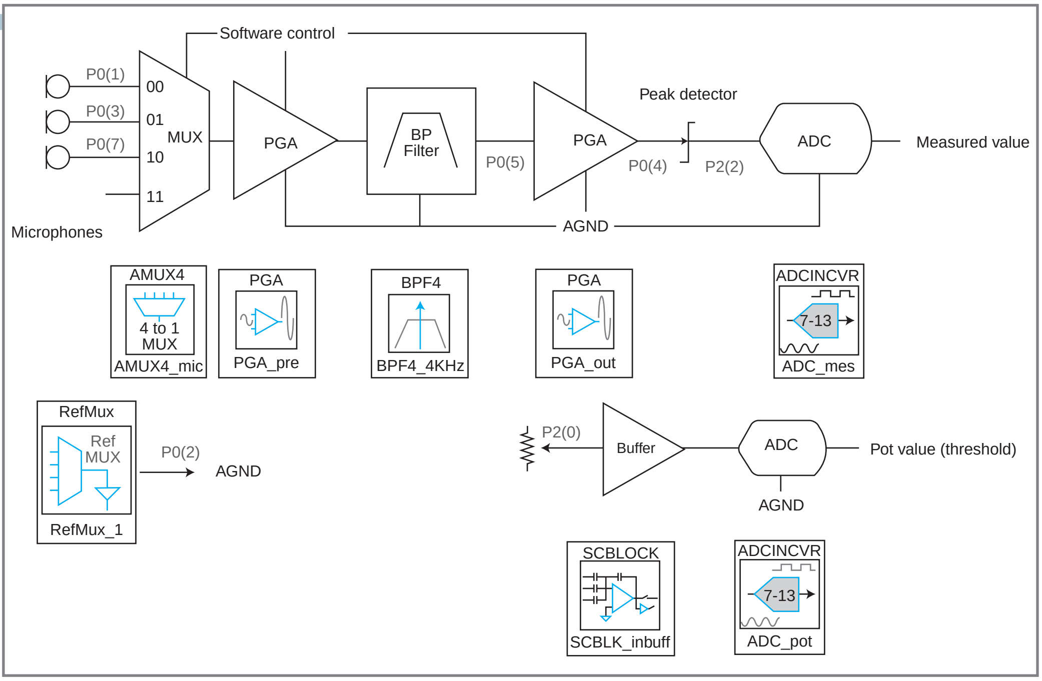


Figure 3—This is equivalent to Figure 2, but it's drawn using standard symbols.

specific I/O pin to the specific input of the block. Substituting a standard MUX with an AMUX4 or AMUX8, you can dynamically switch four or eight pins toward an input during the execution of the program using its API. This module doesn't fill up a block since it superimposes the function of an existing module.

In this configuration, ports P0(1), P0(3) and P0(7) can be sequentially addressed to PGA_pre.

PGA_pre refers to the programmable gain amplifier (PGA). The microphone's signals need a first amplification before entering into the band-pass filter. The ACB00 block accepts analog type-B module, its input is internally connected to the AMUX4_mic and its output can be internally connected to the analog switched capacitor block ASC10 without using any bus. It is perfect for our purposes. The characteristics of this amplifier are good for an audio-band amplifier—GBP up to 5.4 MHz for a 5-V power supply in high-power mode. The gain is programmable in several steps from 1/16 to 48. It is amazing to me to read about GBP, noise factor, bias, etc. in a programmable-device datasheet.

Using an amplifier with a software programmable gain enables us to realize an automatic gain control, expanding the dynamic of the amplification chain. This is another feature not present in the op-amp version of this circuit.

BPF4_4KHz refers to the band-pass filter. This is a

resistors drive the LEDs. A Schottky diode forms a peak detector together with a capacitor to measure the level of sound; this kind of diode causes a lower voltage dropout than normal rectifier diodes. The purpose of the R5 resistor is to decrease the discharge time of the C5 capacitor in order to enable a faster multiplexing of the three inputs. Those external components were chosen instead of a peak-detection block because all the internal available boxes are busy and this three-component network is an acceptable compromise. A trimmer is used to set up the threshold of the analog signal that triggers digital out. A trimmer? Why a trimmer when you can use an up/down digital circuit with an LED bar indicator of the position? Because it's much simpler. Sometimes I'm lazy.

Filtering capacitors and sockets complete the "bill of materials." The real complexity is in the "internal" part of the circuit, as expected using this PSoC device. Let's describe every single block and connection.

ANALOG BLOCKS

Figure 2 shows the analog blocks configuration with the connections to the internal buses and external pins drawn using PSoC Designer schematic conventions. Figure 3 shows the equivalent schematic diagram drawn using standard symbols.

The AMUX4_mic block is one of the improvements with respect to the older op-amp version. In that one, the microphones were all mixed in one single input, not enabling the identification of the sound source direction. With a multiplexer, we can use a single chain of filtering and amplification, assigning it to one microphone at a time.

All the analog continuous blocks (first row) are connected to the external world through a standard MUX or a combination of multiplexers. With PSoC Designer you can address a

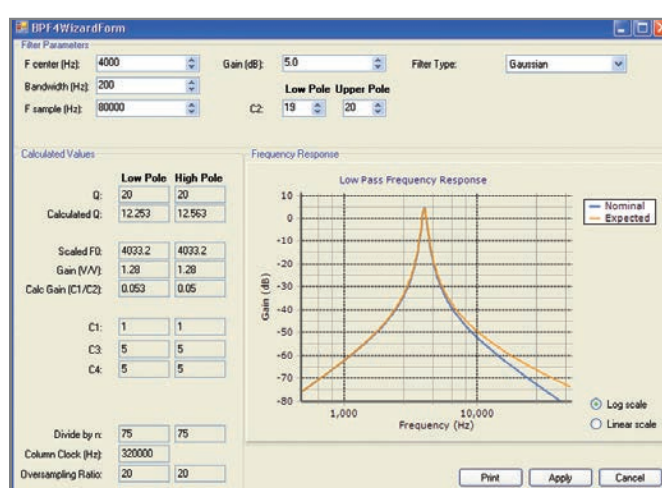


Photo 4—You can use the filter design to write down all the values for all the registers involved in your filter. Simply right click on a filter block to pop up the contextual menu.

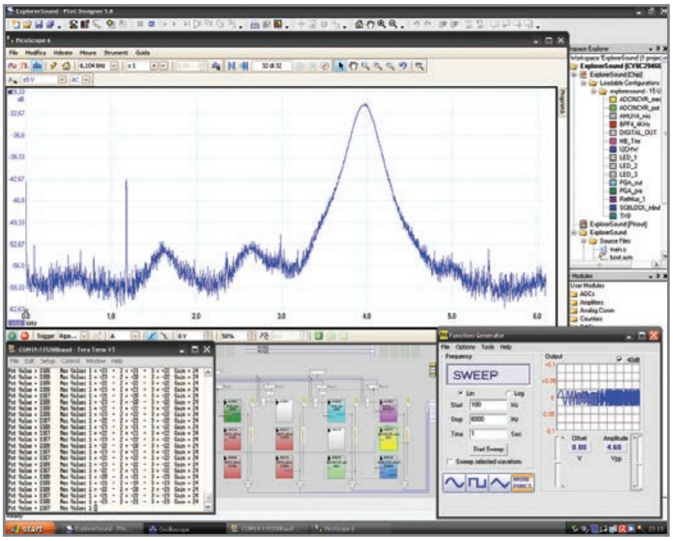


Photo 5—The graph captured by the DSO in Spectrum Analyzer mode, the function-generator window that sweeps a signal in the audio band, and a terminal window with the debugging values sent by the PSoC via the UART port. The background shows the PSoC Designer window with the project used for this test.

switched capacitor module that fills up four blocks. The output of BPF4_4KHz goes to AnalogBus 1 and out to Port_0_5. The Port_0_5 pin is internally connected also to AInMux_2 and routed to PGA_out for the final amplification.

These are the most interesting kind of modules. With switched-capacitor amplifiers you can build fully analog filters with the same characteristics of classic filters (e.g., Butterworth and Bessel).

Second-order filters fill up two switched-capacitor analog blocks (ASD and ASC). Fourth-order filters fill up four blocks. Remember, there are some connection constraints that limit the number of practicable filters. Only second and third rows accept switched-capacitor blocks. Only the first and third columns can be connected to an ACB block, the only one that enables connection to the world external to the device.

Not every filter topology is available on every PSoC 1 family. For example, Cypress's eight-pin CY8C27143-24PXI PSoC device, even with 12 analog blocks, admits only BPF2 and LPF2 filters. All available filters topology are band pass of second order (BPF2), band pass of fourth order (BPF4), elliptical low pass of second order (ELPF2), elliptical low pass of fourth order (ELPF4), low pass of second order (LPF2), and low pass of fourth order (LPF4).

We need some basic theory about the switched-capacitor technique to understand what we are going to do. Switched cap blocks use the same op-amp used in the continuous analog blocks. This op-amp then has some capacitors driven by switching signals placed in feedback, input, and output positions to serve various functions such as integrators, summers, and filters. The timed switching signals give the capacitors the ability to function the same as a resistor would in the circuit. The value of the capacitance along with the speed of the switching enables the user to vary the correlating resistance value.

This is the heart of the switched-capacitor technology. Since switching is involved, the output signal you would expect isn't continuously present and must be sampled at specific times in order to see the correct signal. The construction of the PSoC switched-capacitor blocks enables you to process these signals easily in filters and put them out to the analog output buffers or use the blocks as analog-to-digital converters without being a switched capacitor expert.

Once more, you can write down all the values for all the registers involved in your filter or use the practical filter design wizard by right clicking on a filter block to pop up the contextual menu (see Photo 4). This wizard suggests the capacitance values needed, after entering the characteristics of the filter you are going to realize. The resulting filter frequency response is shown in both theoretical and effective form. As happens in classical discrete component filters, you have to slightly adjust values to obtain the better compromise on frequency bandwidth, phase, and amplitude flatness.

One of the important things to note about designing a filter is that the cutoff frequency of the filter is based on the clock frequency being sent to the analog columns that contain the filter. The filter design wizard does not set up resources to generate this frequency for you. It simply tells you which frequency is important. The specified frequency must be sent to all switched-cap blocks used in the filter. You will be required to create a clock using digital resources to send to the analog column. This clock can come from timers, counters, and so on. If the clock is attainable by using the VC1, VC2, or VC3 signals, you can reserve your digital blocks for other uses. Since there are limits on what clocking frequencies can be set to the blocks, there are limits on what frequencies you are able to choose for your filters. That is the reason why you won't find a high-pass filter design in the PSoC. Unfortunately, this is the nature of switched-capacitor designs.

Photo 5 shows the response, in the frequency domain, of the realized band-pass filter. The screenshot shows the graph captured by the DSO in Spectrum Analyzer mode, the function-generator window that sweeps a signal in the audio band, and a terminal window with the debugging values sent by the PSoC via the UART port. The background shows the PSoC Designer window with the project used for this test.

Another PGA of the same type is PGA_out. This one amplifies the signal at the out of the BPF4_4KHz. It is also software controlled, which enhances the dynamic of the entire chain even more.

PGA_out is the output that is connected to AnalogOutBuf_2 and routed to Port_0_4. The filtered signal is therefore available on Port_0_4. Port_0_4 is connected to Port_2_2 through a Schottky diode and a capacitor in order to rectify the 4-kHz signal. The DC signal out of the rectification circuit on Port_2_2 is routed to ADCINVR_mes to measure via software the level of the 4-kHz signal present at the input.

With the scope placed on Port_2_2, you can see the signal (see Photo 6). The configuration is the same as in Photo 5, with the DSO in Time Domain mode (oscilloscope). The sampling frequency of the switched capacitor block is clearly visible, as previously described.

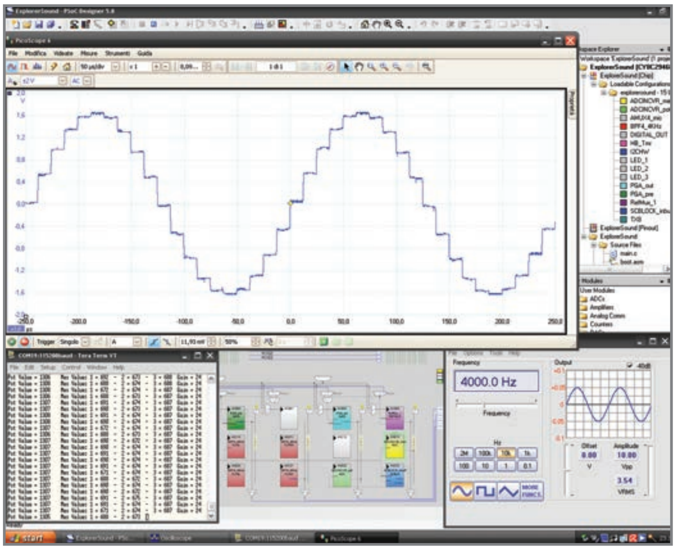


Photo 6—With the scope placed on Port_2_2, you can see the signal. The configuration is the same as in Photo 5, with the DSO in Time Domain mode (oscilloscope).

RefMux_1 refers to the analog virtual ground reference. This is routed to AnalogOutBuf_3 in order to obtain an AGND virtual ground reference for the inputs. It is connected to them through a 1-MΩ resistor. Another interesting module available on PSoC devices, this enables the generation of a virtual ground with a level between VDD and VSS. Anyone who has been involved in analog circuit design with a single power supply knows how much care must be dedicated to this part of the circuit. A bad design can damage the signal introducing distortion or noise. A passive network virtual ground reference can lower down the input impedance at unacceptable values.

Studying several application notes, something caught my attention. They always use a resistor between inputs and AGND reference. Why? All analog blocks should be automatically referenced to AGND internally, what is the purpose of that resistor? On a classic op-amp there is always some feedback resistor to the input that brings it to the right bias. A switched-capacitor amplifier input, decoupled with a capacitor to the external circuit, is really floating with the risk of having an unwanted DC bias or amplifying a noise. It needs a resistor between each input pin and the virtual ground reference to fix the bias.

AGND reference can be realized in different ways. As in normal op-amp, when no more pins are available on your device, you can use a voltage divider with two resistors and a capacitor from VDD (a Cypress Semiconductor AN16833 PSoC). In application note AN2247, Cypress simply uses an unused OutBuff connected to an output pin. If a block and a pin are available, the best way is what you can see on this circuit: configuring a RefMux with AGND as input and OutBuff as output to an external pin, this one becomes a clean and stable AGND reference for all of the circuit.

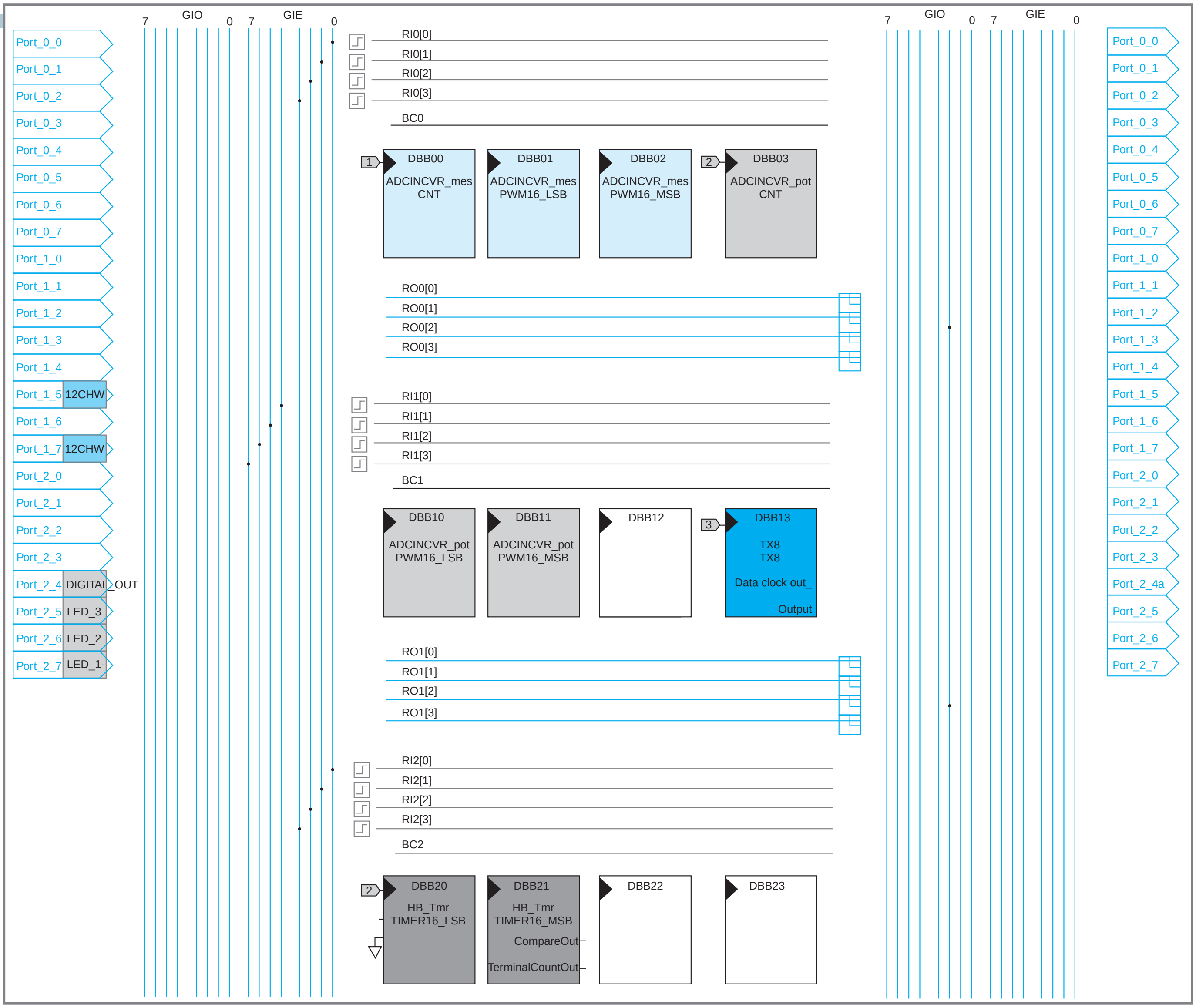


Figure 4—Digital blocks diagram

SCBLK_inbuff, the generic SCBlock, is just used as a buffer with a gain of one to connect the external Port_2_0 with ADCINCVR_Pot ADC. This is a trick that popped up after hours of tests on how to connect the port on which the trimmer is with an ADC that can only be placed on a fourth column because the other columns are already busy. It was also a good exercise for understanding an SCBlock. I had undervalued this generic block because it was too "generic," and also because the datasheet is too poor.

After looking at application note AN16833, where a four-channel mixer with independent adjusting for each channel is realized with a single PSoC 1, I found the AN2223 that better explains this module, with relations to standard op-amp characteristics. It's very useful. It's a real op-amp, programmable in several different modes.

Someone once said, "For those who are really adventurous, Cypress has created the generic switched cap block. This block gives you a visual configuration of all the settings of a switched capacitor block to allow you to create your own module easily." It's partially true, indeed. The placement is not trivial, not every connection is possible. There are several restrictions depending on the placement of the block and sometimes some external links are needed, involving an analog bus. But it's really versatile. Unlike classic op-amps, you can also have two inverting inputs.

ADC_mes are, of course, mixed blocks. Part of their function is in the analog section and part is in the digital section. There are many different kind of ADCs available on a PSoC 1. Choosing the one right for your purposes can be tricky. Fortunately, Cypress's documentation is exhaustive with datasheets and a lot of application notes. Discussing in detail the reasons for choosing one or the other is beyond the scope of this article. For this purpose, an ADCINVR with 11-bit resolution is the right compromise between needs and block occupation.

ADCINCVR_mes is used to read the input signal as aforementioned. The ADCINCVR_pot is used to convert the position of a trimmer in order to manually (and optionally) set a threshold for the trigger part of the circuit.

According to the formulas in the datasheet, both types of ADCINCVR_ perform a full measure in 1.7 ms with a 4.8-MHz clock (VC1). These parameters are chosen to have an integration time that filters out some undesired signals. A spreadsheet for calculating parameters is available on the *Circuit Cellar* FTP site.

Like any other analog circuit, and because the maximum possible amplification gain is quite high, it is highly recommended to have a very clean power supply to avoid amplification of noise that could interfere with the real signal. A dedicated linear-voltage regulator or an LC low-pass filter could be used to filter out undesired signals, primarily if the sound circuit is on the same power line of digital circuits.

DIGITAL BLOCKS

Figure 4 shows the digital blocks diagram. These modules are probably more familiar to standard programmable device users. The diagram also uses not so standard conventions to show how the digital modules are connected to the internal buses or to the external pins. This is due to the "not so standard" way that Cypress approaches this issue and to the huge amount of possible connections the PSoC has. This could be a little bit confusing for designers used to working with other kinds of microcontrollers but, after some practice, it quickly becomes easy to manage.

Some boxes in the diagram are filled with the digital counterpart of the ADCs.

The HB_Tmr is a 16-bit timer. It fills up two boxes because it is a 2-B timer. It is the base clock for all of the timings in the program. Its timing is:

$$VC2 = \frac{VC1}{15} = \frac{\frac{SysClk}{5}}{15} = \frac{\frac{24\text{ MHz}}{5}}{15} = 320\text{ kHz}$$

$$\text{Timer Period} = \left(\frac{1}{VC2}\right) \times \left(\text{PeriodRegister} + 1\right) =$$

$$0.000003125 \times \left(3{,}199 + 1\right) = 10\text{ ms}$$

TX8 refers to the serial transmission module. It enables transmission via the UART of the analog values read from various inputs. It can be useful for debugging purposes using standard ASCII characters to a serial console or, using any kind of protocol (e.g., dsNav communication), it could be used to interface other boards alternatively to I^2C port.[1]

It is notable that even only the TX part of the UART module can be used if no data has been received, as in our case. This saves a box and an I/O pin that can be used for other purposes.

Here is some information about digital OUTs. To drive an LED or a generic I/O it is highly recommended to use an LED module. This doesn't fill up a box, it is just a kind of library that correctly drives the I/O port. It can be found under the MiscDigital section. Not using this module requires a direct control of the port using the shadow registers. The module can be configured in Active High or Active Low mode. The first mode must be used when the cathode of the LED is connected to VDD and a high level on the anode will switch it on. The second mode requires an LED with the anode connected to VSS and a low level to the cathode will switch it on. In both cases, the "LED_On" instruction is used to switch the LED on.

The I/O ports can be used even if they are not connected to a block, after all, this is a microcontroller too. A pin configured as "StdCpu" can be used in the program using a mnemonic name.

Every line can be configured in different modes, which enables a lot of flexibility. They accept Strong-high, Strong-low, High-impedance, Pull-up, Pull-down, Open-drain-high, and Open-drain-low modes. Almost everything is possible, even configuring a pin in both Input and Output mode. This requires good practice with PSoC that can be achieved in the field or by a carefully reading many application notes.

In a single PSoC family, there are many kinds of different devices with a different number of I/O pins, but all with the same number of analog and digital blocks. All other pins can be used as input/output like any standard microcontroller. All the outputs can directly drive devices that require several milliamps, such as LEDs or something similar.

EzI2Cs refer to the I^2C communication module. This doesn't fill up a box, you have just to assign a clock and data line to a couple of pins to start using this protocol.

The basic circuit described above performs a basic task, but this approach simplifies much of the design if compared with an equivalent fully analog circuit. In this article, there is a full description of how this specific board works, but there is also most of the information you need to start using PSoC devices when developing similar circuits. The complete and up-to-date PSoC Designer project is available on my Google Code space.

HARDWARE OR SOFTWARE?

It may sound bizarre defining software to hardware differences in a device completely configured through a computer. You are changing some physical parameters via a graphical user interface (GUI) on a computer window. In the same way you can change the flow of a program with a conditional jump, you can optimize the phase response of a Butterworth filter—without a screwdriver on a trimmer or a solder iron on a capacitor—just with mouse and keyboard. And it is not a simulation, it is real world.

But let me be conservative to better explain this innovative device. Let's use traditional terms. In the first part of this article series, I described the hardware part of the project, how the component can be mounted on a printed circuit board (PCB) or on a development board, and how the internal hardware modules can be configured to manage our analog signal in order to be useful for our needs.

Now you have the correct signal at a correct level. What can you do with it? Let's apply some "real" software, at least according to the traditional description of this term.

In the second part of this series, I'll describe the software capabilities of the PSoC 1 device and how to use them to measure the level of the signal, expand the dynamic of the amplifying chain, and reliably share this information in several different modes with the external world. ■

Guido Ottaviani (guido@guiott.com) has made electronics and ham radio his hobby since the "tube times." He turned the hobby into a job working as an analog and digital developer for several years for an Italian communication company. Many years ago, a big change made him a technical manager at a company that develops and manages graphic, pre-press and press systems, and technologies for a large Italian editorial group that publishes sports newspapers and magazines. Some years ago, he got the scope and the soldering iron out of the drawer to make autonomous robots. Currently, Guido is an active member in various Italian robotics groups who shares his experiences with other self-professed "electronics addicts" and evangelizes amateur robotics.

PROJECT FILES

To download the "ADCINVRcalc .xls" spreadsheet for the calculation of parameters, go to ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2011/256.

REFERENCE

[1] dsNav Communication, www.guiott.com/Rino/CommandDescr/Protocol.htm.

RESOURCES

Cypress Semiconductor Corp., "PSoC 1 Overview," 2011, www.cypress.com/?id=1573.

———, Cypress PSoC 1 Application Notes Database, www.cypress.com/?app=search&searchType=keyword&keyword=&rtID=76&id=1573&applicationID=0&l=0.

Google, Explorersound, a PSoC-based system for 4-kHz tone decoder, http://code.google.com/p/explorersound.

G. Ottaviani, "Robot Navigation and Control (Part 1): Construct a Navigation Control Subsystem," *Circuit Cellar* 224, 2009.

———, "Robot Navigation and Control (Part 2): Software Development," *Circuit Cellar* 225, 2009.

———, "A Sensor System for Robotics Applications," *Circuit Cellar* 236, 2010.

SOURCES

CY8C29x66, CY8C29466-24PX, CY8C27143-24PXI, AN1683, and AN2223 PSoCs, CY3210-PSoCEval1 and CY3217-MiniProg1 evaluation kits, and PSoC Designer (IDE)
Cypress Semiconductor Corp. | www.cypress.com

NE567 Tone detector
NXP Semiconductors | www.nxp.com