Introduction

Process and environment variations conspire against radio frequency and analog mixed-signal (RF/AMS) designers by adversely affecting die performance, yield, and lifespan. Designers are faced with a classic tradeoff: design margins can be increased to mitigate these effects, but too often at steep power and area penalties. In addition to these challenges, ever-increasing levels of silicon integration demand complex testing of RF/AMS components for product qualification. [1] Die testing is indispensable, yet expensive, particularly for RF/AMS circuits due to the need for specialized automated test equipment (ATE). Reduced testing can lead to increased levels of escapes—undetected products with failures—that can exponentially increase costs as silicon products traverse the integration chain.

Meanwhile, as the cost of CMOS digital component and testing costs diminish by comparison, designers are beginning to consider hybrid solutions to RF/AMS problems that incorporate digital calibration and compensation, and test circuitry. Embedded digital signal processors (DSPs) are now sufficiently small to serve as viable options for complementing RF/AMS cores. Two IEEE Journal articles [2] [3] propose embedded DSPs perform a variety of tasks (such as listed in Table 1 and Table 2) with a series of goals:

- Extend silicon life
- Simplify the digital design process
- Optimize test coverage and costs
- Increase manufacturing yields, reduce current consumption, and adapt to varying environmental conditions [3]

Programmable DSPs reduce the risks associated with developing a complex system-on-chip (SoC) where all problems cannot be predicted in the pre-silicon stage. Programmability allows for post-silicon upgrades and fixes, potentially saving silicon re-spins and supporting aggressive product cycles.
Table 1: DSP-Assisted RF/AMS Calibration Tasks

<table>
<thead>
<tr>
<th>Component</th>
<th>Impairment</th>
<th>DSP Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>RF transceiver</td>
<td>• IQ imbalance</td>
<td>• Estimate impairment and calibration parameters</td>
</tr>
<tr>
<td></td>
<td>• DC offset</td>
<td>• Adjust analog components (e.g., tuner LO frequency)</td>
</tr>
<tr>
<td></td>
<td>• Carrier offset</td>
<td>• Digital compensation (e.g., DC offset removal)</td>
</tr>
<tr>
<td></td>
<td>• In-band ripple</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Spectral mask violations</td>
<td></td>
</tr>
<tr>
<td>Analog-to-digital converter (ADC)/Digital-to-analog converter (DAC)</td>
<td>• Nonlinear distortion</td>
<td>• Optimize calibration parameters (e.g., programmable gains, capacitors)</td>
</tr>
<tr>
<td></td>
<td>• Missed codes</td>
<td>• Optimize/apply correction filters (e.g., Volterra filter)</td>
</tr>
<tr>
<td></td>
<td>• Poor effective number of bits (ENOB), signal to noise and distortion ratio (SINAD), spurious free dynamic range (SFDR)</td>
<td></td>
</tr>
<tr>
<td>High-speed parallel ADC</td>
<td>Non-uniform timing</td>
<td>Measure and compensate timing delays</td>
</tr>
<tr>
<td>SerDes PHY</td>
<td>• Voltage and temperature variations</td>
<td>• Optimize parameters with a high-level state machine</td>
</tr>
<tr>
<td></td>
<td>• Frequency selective channels: silicon, package, and PCB variation</td>
<td>• Generate training for equalizer adaptation</td>
</tr>
</tbody>
</table>

Table 2: DSP-Assisted RF/AMS Testing Tasks

<table>
<thead>
<tr>
<th>DSP Function</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spectral analysis (FFT)</td>
<td>Measure ADC/DAC ENOB, SINAD, SFDR</td>
</tr>
<tr>
<td>Histogram</td>
<td>ADC/DAC integral and differential nonlinearity (INL/DNL), ENOB</td>
</tr>
<tr>
<td>Arbitrary signal generation</td>
<td>Probe RF/AMS blocks with arbitrary signals (e.g., sinusoid, ramp)</td>
</tr>
<tr>
<td>Error vector magnitude (EVM), bit error rate (BER)</td>
<td>Replace vector spectrum analyzer (VSA) and other high-cost test equipment</td>
</tr>
</tbody>
</table>

Advanced and robust solutions are necessary to tackle design and verification challenges in hybrid designs that consist of RF/AMS and digital components. Cadence serves the market with advanced tools, flows, and methodologies that help analog designers overcome these challenges. Cadence Virtuoso and Incisive® tools, such as Cadence Virtuoso AMS Designer, SimVision™, and vManager™ solutions, provide a complete design and verification solution for analog, RF, mixed-signal, memory, and SoC designs. The Cadence integrated mixed-signal (MS) implementation solution offers a variety of capabilities to circuit and verification engineers, including:

- Ability to browse design hierarchy to view signals and design details
- Ability to interchange from transistor-level to behavioral-level blocks
- Integrated analog and digital solvers
- Intuitive interactive debugging platforms, which is essential for applying debugging techniques
- High performance for MS verification

This paper discusses a case study of a pipeline ADC [4] that is integrated with an embedded Cadence Tensilica processor. The processor is used to correct typical ADC non-idealities that decrease performance parameters such as signal-to-noise-and-distortion-ratio (SINAD), spurious free dynamic range (SFDR), differential and integral nonlinearity (DNL and INL), and effective number of bits (ENOB). The AMS circuitry and Tensilica processor are integrated and modeled in the Cadence Virtuoso AMS Designer and Xtensa Xplorer development environments, providing a high level of integration and compatibility and a simplified design and verification process.
This document includes a brief overview of Tensilica processors and the ConnX D2 DSP used in this case study. It reviews the structure of the pipeline ADC and the types of non-idealities that may be encountered in this type of mixed-signal block, and discusses the application of the processor to estimate the parameters for pipeline ADC error correction. The Virtuoso AMS Designer environment is used to design and test the system, and the Eclipse-based Xtensa Xplorer integrated development environment (IDE) is used for processor configuration and customization.

**Cadence Tensilica Processors**

Cadence Tensilica processors can be optimized to run a class of applications very efficiently, striking a balance between die area, energy, and performance that is closely matched to the tasks to be performed. Starting with the baseline Xtensa architecture, designers can add VLIW (very long instruction word), SIMD (single instruction multiple data), DSP, and custom instructions to speed up critical operations. The Xtensa Xplorer IDE allows customers to profile their code and develop their own processors, or to select a ready-to-use processor designed by Cadence. These processors range widely in performance and size, from 32-bit embedded controllers to 128-way SIMD/VLIW vector processors. They are all supported by a single C/C++ software development tool chain, with application-specific libraries and third-party ecosystems matched to the end use. Table 3 lists Tensilica DSP families that can be included in Tensilica processors, and the common types of applications for which they are optimized.

**Table 3: Cadence Tensilica DSPs**

<table>
<thead>
<tr>
<th>Tensilica DSP Family</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fusion</td>
<td>Multi-functional DSP for IoT</td>
</tr>
<tr>
<td>HiFi</td>
<td>Audio / voice / speech</td>
</tr>
<tr>
<td>Vision</td>
<td>Imaging / computer vision</td>
</tr>
<tr>
<td>ConnX</td>
<td>Communications / baseband</td>
</tr>
</tbody>
</table>

If a different design balance is needed for a specific testing/calibration application, an engineer can use Xtensa Xplorer to make adjustments to lower cost or increase throughput, such as changing the memory configuration or multiplier hardware size, or adding a second load/store unit or a single or double-precision floating-point unit. The Xplorer IDE is also used for software development, so that hardware and software engineers have the same views of the design as it progresses from architectural exploration to post-silicon application development.

**Tensilica DSPs for Mixed-Signal Testing and Calibration**

Mixed-signal testing and calibration makes extensive use of digital signal processing. In our case study, advanced DSP functions are used for least-squares parameter estimation and spectral analysis of signal quality. The Tensilica ConnX D2 DSP (Figure 1) is a small general-purpose fixed-point DSP that is a good initial fit for this application.

[Diagram of ConnX D2 DSP Architecture]

However, with the recently released Fusion F1 DSP, we recommend it as the starting point for new designs. The Fusion F1 DSP offers up to four-way single instruction/multiple data (SIMD), in this case four 16-bit MAC operations per cycle, compared to the two-way SIMD in the ConnX D2 DSP. The ConnX D2 and Fusion DSPs are
compared in Table 4. Using the Xtensa Xplorer options, these DSPs can be included in a Tensilica processor with a click, and further optimization is still possible. Designed to be programmed in C with ITU-T/ETSI-compatible intrinsics, these DSPs can sustain a throughput of two or four saturating fixed-point multiply-accumulate (MAC) operations per clock through a combination of SIMD and multi-issue flexible length instruction extensions (FLIX) operations, with a loop buffer to eliminate branching overhead. Real and complex vector operations and specialized circular and bit-reversed address modes support fast Fourier transform (FFT) and other transform algorithms. Fixed-point algorithms are fast and efficient, but have limited dynamic range compared to floating-point implementations. To give the programmer the ability to choose freely between fixed- and floating-point implementations, the Tensilica processor configured for this case study includes a single-/double-precision IEEE 754-compliant floating-point unit that adds floating-point registers and ALUs with throughput of one add, subtract, multiply, or fused multiply-add operation per cycle.

Table 4: ConnX D2 and Fusion DSP Comparison

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>ConnX D2 DSP</th>
<th>Fusion F1 DSP</th>
</tr>
</thead>
<tbody>
<tr>
<td>MACs per cycle</td>
<td>2</td>
<td>2 or 4</td>
</tr>
<tr>
<td>SIMD width</td>
<td>2-way</td>
<td>2- or 4-way</td>
</tr>
<tr>
<td>VLIW</td>
<td>Dual issue (optional 3-way issue)</td>
<td>Dual issue</td>
</tr>
<tr>
<td>Load / store</td>
<td>Single or dual 32-bit</td>
<td>Single 64-bit</td>
</tr>
</tbody>
</table>

**Pipeline ADC and Error Modeling**

A typical pipeline ADC consists of a cascade of multiple sub-ADC stages, each stage extracting a few bits of overall ADC resolution at a time [4]. Figure 2 illustrates a 12-bit pipeline ADC example with a 3-bits-per-stage architecture. Each stage consists of a sample-and-hold (S/H), a 3-bit flash ADC, a 3-bit DAC, and an amplifier (excluded in the last stage). The most significant bits (MSBs) are converted by the first stage. The remaining bits are converted by the stages that follow until the least significant bits (LSBs) are converted by the last stage. The input of each stage is sampled by the S/H block and the sampled output \( V_{in} \) is sent to the flash sub-ADC that resolves 3 bits. This 3-bit output \( D_i \) is converted back to an analog signal through the DAC. The “residual signal” \( V_{out} \), which is the DAC output subtracted from the S/H output \( V_{in} \), and amplified by the gain of \( G \) (nominally \( G = 8 \) for a 3-bit stage), feeds the next stage of the ADC until 12 bits have been resolved. Due to data latency that is introduced from the time the analog input is first sampled until the LSBs have been resolved by the last stage, a time alignment module is added to synchronize and concatenate all stage output bits corresponding to the same sample.

![Figure 2: 12-Bit, 4-Stage Pipeline ADC](image-url)
The nominal output $V_{out}$ of each stage is given by

$$V_{out} = G \cdot V_{diff} \tag{1}$$

$$V_{diff} = \left[V_{in} - \frac{(di-3.5)W_{ref}}{4}\right] \tag{2}$$

Non-idealities in sub-ADC stages can introduce errors to ADCs, reducing performance and accuracy. Different types of errors occur in the pipeline stages, mainly due to sub-ADC comparator offsets, operational amplifier (OPAMP) non-linear gain with signals, capacitor mismatches, OPAMP finite gain and bandwidth, and OPAMP offset. These cause residual gain and offset errors in each stage that manifest as integral and differential non-linearities (INL and DNL) errors in the ADC and show up as harmonic or intermodulation distortion components in the ADC output spectrum.

**Sub-ADC Stage Error Modeling**

The non-linear gain error and OPAMP offset are introduced in the amplifier of each stage modifying the ideal residual output equation (1). Parameters $\alpha$, $\beta$, and $V_{os}$ model the non-idealities as follows:

$\beta$ models the constant gain error of the residual amplifier by changing gain $G$ to:

$$G = G_{ideal}(1 - \beta) \tag{3}$$

$\alpha$ models the gain compression error in the residual amplifier by changing (1) to:

$$V_{out} = G\left(V_{diff} - \alpha V_{diff}^3\right) \tag{4}$$

$V_{os}$ models the offset voltage of the residual amplifier by changing (4) to:

$$V_{out} = G\left(V_{diff} - \alpha V_{diff}^3\right) + V_{os} \tag{5}$$

For an ideal ADC, the residual waveform of each stage is always within the range of the next sub-ADC stage. However, the contribution of gain and offset errors will force the residual waveform to go outside of the range, leading to output code errors. These incorrect codes will be added in the overall ADC transfer function. For a severely impaired pipeline ADC, the digital output “sticks” on a code and “jumps” over an output region whenever the residual waveform is outside of the sub-ADC range [5] (see Figure 3). Additionally, the transfer curve can display non-monotonic behaviour for some ADC impairments when codes “dip.”

![Figure 3: ADC Transfer Function with Missing Codes Due to Over-Range Error](www.cadence.com)
Differential Non-Linearity (DNL) Error

Differential non-linearity (DNL) describes the deviation between the ideal analog input step-size to the actual step-size observed for each ADC code. The ideal step-size is 1 LSB. This deviation is represented in units of LSBs. In Figure 4, DNL is measured as –0.6 LSBs at code 14 and +0.25 LSBs at code 19. Usually, the maximum value of the DNL curve is quoted for an ADC instead of the whole DNL curve [6].

![Figure 4: Differential Nonlinearity](image)

Integral Non-Linearity (INL) Error

Integral non-linearity (INL) describes the deviation of the actual analog input to digital output code transfer function with respect to the ideal transfer function for each ADC code. This deviation is represented in units of LSBs. By definition, INL for a particular code is the summation of the DNL array elements up to that code. Usually the maximum value of the INL curve is quoted for an ADC instead of the whole INL curve. Figure 5 shows two sets of curves. The straight gray line is the ideal transfer function. The quantized red curve represents an ideal ADC output. The INL for this ideal ADC is 0.5 LSB. The quantized blue curve represents the output of an ADC with distortion. The INL error here is greater than 0.5.

![Figure 5: Integral Nonlinearity](image)
Pipeline ADC Calibration via an Embedded Cadence Tensilica Processor

This case study demonstrates the calibration and testing of a pipeline ADC via an embedded Tensilica processor that includes the ConnX D2 DSP. Note that we recommend using the recently released Fusion F1 DSP as the starting point for new designs.

The methodology estimates and corrects ADC nonlinearities applied to an AMS SoC. The case study uses Cadence Virtuoso and Incisive tools for modeling AMS circuitry and solving SoC-level mixed-signal verification challenges, and Cadence Xtensa Xplorer for processor configuration and emulation. A 12-bit, 20Msps pipeline ADC based on a subranging architecture is followed by the Cadence Tensilica ConnX D2 processor and a digital correction filter to estimate and correct the ADC errors, respectively. See Figure 6. A non-linearity gain error is introduced in the amplifier block of each sub-ADC stage modeling the effect of capacitor mismatch, finite op-amp gain, and various switch-induced offsets. Therefore, the actual amplifier gain G is different than the ideal (G_{ideal}=8). The gain and the residual output of the amplifier are calculated using equations (3) and (4). The gain compression error $\alpha$ and constant gain error $\beta$ have been chosen as 2% and 1% respectively.

![Figure 6: Pipeline ADC Architecture with Embedded Tensilica Processor](image)

This example presents two scenarios:

- A sinusoid signal is generated externally and converted to a digital signal through the pipeline ADC. A processor is added to calibrate ADC non-idealities, which are corrected through the filter.

- A digital signal is generated internally by the processor and is converted to analog through the DAC. The analog signal feeds the ADC and is converted again to digital. The ADC non-linearities are calibrated and corrected through the processor and compensation filter.

An analog multiplexer (MUX), controlled by the processor, is used to select the input of the ADC as either an external signal or internal loopback signal. The external signal can be either a calibration stimulus (such as a sinusoid) or the “mission mode” signal for regular operation of the SoC. The internal loopback signal is generated by the processor for calibration and testing.

The correction filter can be included as software running on the processor or as a hardware block outside of the processor. The hardware implementation of the correction filter saves power as the processor can be shut off while the filter is in operation. Alternatively, the processor can be used for other tasks such as calibration of other RF/AMS components on the SoC.

---

1 $\alpha$: Parameter representing the sub-ADC stage’s gain compression (with signal). This parameter is calculated from the OPAMP’s 1dB compression point, translating it to the sub-ADC’s compression point and then to its gain error. An error of 2% is a typical number for 50-60dB gain circuit.

2 $\beta$: Constant gain error uncertainty that occurs mainly due to capacitive mismatch. We can compensate for a capacitive mismatch as low as 1% (7- to 8-bit matching).
The analog signal is converted to a 12-bit signal through the 4-stage pipeline ADC. The ADC begins sampling data (sampling rate 20Msps) and storing it to the FIFO. After the FIFO is loaded with 4,096 samples, an interrupt is sent from the FIFO controller to the processor announcing that the data is ready. The processor reads the data and starts performing least squares (LS) estimation of the digital compensation parameters [7], which is completed in under 500K cycles. Assuming a clock rate of 400MHz, processing time is less than 1.25msec. Once the parameters are derived, they are programmed by the processor into the compensation filter, which corrects the distortions in the digitized signal. At this point, calibration is complete and the system can start processing “mission mode” data.

Parameter estimation and digital correction using the processor and the compensation filter is illustrated in Figure 7. The processor is modelled in SystemC. Initially, the processor applies a phase-locked loop (PLL) to the distorted output samples of the ADC and generates a synthetic tone that is locked in frequency and phase to the ideal input tone. The synthesized ideal signal along with ADC calibration samples are subsequently passed to a digital processing block to form LS equations, which estimate the coefficients that feed the compensation filter. The processor solves the LS problem by performing Cholesky decomposition of the observation matrix followed by forward/backward substitution. The PLL is allowed to run for 4,096 samples and the last \( L = 1024 \) samples are used for the LS estimation. After coefficient estimation, the processor is shut off to conserve power.

The digital correction algorithm that is used in this example improves important ADC dynamic performance measures, such as SINAD, SFDR, and ENOB as shown in Table 5. All of these metrics are calculated directly on the processor by applying an FFT to the distorted and compensated signals. Figure 8 and Figure 9 shows the distorted pipeline ADC output and post digital correction signal spectrums, respectively. As Table 5 shows, SINAD, SFDR, ENOB, and ADC INL errors are significantly reduced.

The case study shows the clear advantages of using a Tensilica processor with the ConnX D2 option for calibration and testing. Its fixed-point dual MAC unit is used to efficiently implement the PLL. The data collected from the PLL is further processed by the dual MAC to generate the LS inner product matrices (for example, \( A' A \) and \( A' b \) for \( Ax = b \)). Meanwhile, the Cholesky decomposition (for example, \( A' A = LL' \)) and forward (e.g., \( Ly = A' b \)) and backward (e.g., \( L' x = y \)) substitution steps are easily implemented using the processor’s floating-point unit.
Table 5: ADC Dynamic Performance Measures

<table>
<thead>
<tr>
<th>Performance Metric</th>
<th>Distorted Signal</th>
<th>Compensated Signal</th>
<th>Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>SFDR (dB)</td>
<td>62.1</td>
<td>86.8</td>
<td>24.7</td>
</tr>
<tr>
<td>SINAD (dB)</td>
<td>57.9</td>
<td>71.0</td>
<td>13.1</td>
</tr>
<tr>
<td>ENOB (bits)</td>
<td>9.3</td>
<td>11.5</td>
<td>2.2</td>
</tr>
<tr>
<td>INL range (LSB)</td>
<td>(-5,+5)</td>
<td>(-1,+1)</td>
<td>4</td>
</tr>
</tbody>
</table>

Virtuoso AMS Designer Solution Using Real Number Models

In this example, the pipeline ADC is modeled in Verilog-AMS using wreal (real wire) real-number modeling (RNM). Cadence has developed a Verilog-AMS-based RNM solution that allows high-performance and reasonably accurate modeling of analog behavior to aid verification of mixed-signal designs. With RNM, an analog block can be described as a signal-flow model. Real numbers can represent voltage and current level, and real signals change values based on discrete events [8]. Therefore, the simulation is executed by a digital simulation engine without needing analog solvers. This results in high simulation performance—the simulation is nearly as fast as pure digital simulation, making RNM essential for large mixed-signal SoC verification. Less modeling effort is required to develop RNM than a traditional analog behavioral model—and because there are no analog engines, convergence issues are avoided.

Mixed-signal verification becomes more and more difficult as more design and verification challenges are introduced in mixed-signal designs. Cadence tools such as Virtuoso AMS Designer, Virtuoso Analog Design Environment, Virtuoso Visualization and Analysis (ViVA), Incisive vManager, and SimVision provide a complete design and verification solution for analog, RF, mixed-signal, memory, and SoC designs. Cadence Virtuoso AMS Designer Simulator supports both top-down and bottom-up design flows with the ability to interchange different levels of abstraction. Design levels can change from transistor-level to behavioral-level. This solution provides comprehensive behavioral language support for Verilog-AMS and other languages such as Verilog-A, VHDL-AMS, System Verilog, etc. Verilog-AMS wreal models, as well as SV-RNM, are fully supported.

Cadence Virtuoso AMS Designer is a robust mixed-signal verification solution based on analog and digital simulation engines integrated in a single kernel. Virtuoso Analog Design Environment XL, along with the Virtuoso AMS Designer simulator, provide an advanced design and simulation environment. In Virtuoso Analog Design Environment XL, you can simulate multiple testbenches simultaneously to verify in a single run the performance across multiple tests. You can also run multiple simulations using sweeps or corners to parameterize multiple variables, model files, etc. Specifications can be set up for specific expressions and indicators are given both in words (pass/fail/near) and in color (green/red/yellow) showing whether the values of the expressions are less than, greater than, or near specifications. Virtuoso Analog Design Environment XL also supports Monte Carlo analysis,
which is useful for verifying how robust the design will be in production [9]. Figure 10 shows an example of how the results are presented in Virtuoso Analog Design Environment XL after running a simulation using corners and specifications.

Virtuoso AMS Designer links the Cadence Virtuoso custom design platform with the Cadence Incisive functional verification platform [10]. Cadence SimVision debug solution is essential for mixed-signal verification. It provides a unified graphical environment for the Virtuoso AMS Designer simulator and other Cadence simulators. The SimVision debug solution is made up of several debugging tools, each with its own window. The Design Browser (Figure 11) lets you navigate the design hierarchy, access the objects in the design, and monitor signal value changes during and after the simulation. The Source Browser (Figure 12) provides access to the source code, and lets you set breakpoints and run forward or backward through the simulation, starting at any point in simulation time. The Waveform window (also shown in Figure 12) lets you view software and hardware states, such as structures, fields, and variables, as well as design under test (DUT) analog, real, and logic signals.
This case study uses SimVision post-processing features for debugging. The Source Browser fully supports Verilog-AMS with RNM, which aids in monitoring real signals and variables values. Figure 13 shows the SimVision Waveform window, displaying the outputs of an ideal ADC (ideal_signal), an ADC with non-idealities (distorted_signal), and the ConnX D2 DSP, as well as the correction filter (corrected_signal) correcting the ADC non-idealities. The distorted_signal includes the non-linearities that were introduced in each stage of the pipeline ADC and cause the distortion of the ADC output. ConnX D2 DSP, along with the correction filter, corrects those non-linearities, restoring the distorted signal and providing the corrected signal as output. As Figure 13 shows, corrected_signal and ideal_signal are almost identical signals since the DSP restores the distorted signal.

Optimizing an Xtensa Processor Using Xplorer

Cadence Xtensa Xplorer provides a single environment for processor and software design, enabling processor architects and programmers to work in the same design environment, using the same tools, making it easier to work together through the entire project design cycle to produce a highly efficient product. Xplorer can be used to configure Tensilica processors and to build and debug software so that users have full-project management for the system and the software in a single platform. The Xplorer IDE is a full-featured software development workbench that includes tools for software development, simulation, profiling, debugging, hardware trace, processor configuration/extension, and building variants of software tools. For this case study, Xtensa Xplorer is used to configure and build a Tensilica processor with the ConnX D2 DSP and a floating-point unit. The same Xtensa Xplorer SDK is used to develop and debug the software.
Step 1: Select Configuration Options

The first step for processor optimization is to select the configuration options that will be included in the processor and the software development tools. Figure 14 shows some of the processor configuration options available, such as DSPs and floating-point/math hardware. Other configuration options such as local memories, bus interfaces, interrupts, and direct I/O and Queue interfaces are also available. The Queue interface in particular is useful in connecting the processor directly to the AMS components, bypassing the need for complex and power-hungry system bus implementations. If click-box options do not yield enough optimization, Xplorer can be used to define new instructions, registers, states, and ALUs to reduce clock cycles and increase data throughput for specific operations.

![Figure 14: Example Processor Configuration Options in Xtensa Xplorer](image)

Step 2: Write Software and Benchmark

Once the processor has been built, designers need to write software and benchmark it against performance requirements. Users typically develop their code in C/C++ and debug it at the C/C++ source and assembly instruction levels.

Figure 15 shows a few of the Xplorer window options that a user can select to open within the IDE to easily develop software. For example, Xplorer provides the following window options:

- **Project tree view** to organize code
- **Console view**
- **Memory view**
- **Syntax-aware editor**
- **Code browsing with graphically showing the breakpoints and the current line executed**
  - C/C++ source view
  - Assembly code view
- **Contextual help**
- **Debug view** that organizes all debug sessions currently running with control of each debug session
- **View** with list of local variables, all breakpoints, expressions evaluated, processor registers, memory contents, etc.
Step 3: Optimize Desired System Parameters

C code usually delivers good performance on Tensilica processors, but performance can be limited by resource conflicts or memory latency. Figure 16 shows one of the Xplorer tools that helps identify what is limiting program performance. A graphical view of the processor’s pipeline state is correlated with the instructions, spotlighting pipeline “bubbles” where the processor has been stalled so that programmers can rearrange the code and data to reduce pipeline stalls. Xplorer also supports typical software profiling, stack tracing, and code utilization reporting.

Xplorer also estimates power, area, and processor clock speed based on the selected configuration (see Figure 17). If the performance, die area, or power consumption of the processor does not meet the requirements, the processor configuration can be iterated quickly to change features and target speeds.
Conclusion

Applying an embedded processor to digital correction and testing of a pipeline ADC brings the benefits of programmability to an AMS design. A Tensilica processor can be configured with a small DSP and a floating-point unit to implement advanced signal processing algorithms to generate input stimulus signals, estimate digital correction parameters, and measure the resulting ADC performance metrics. Results show a significant performance improvement for the pipeline ADC.

Including a processor in an RF/AMS design has many other advantages, spanning all phases of a product’s life. Starting at the inception phase, the processor can be used to tighten design margins by compensating for process, voltage, and temperature (PVT) variations to save on power and area. The processor can be optimized further to efficiently perform specialized functions that accelerate digital calibration and testing of RF/AMS components.

During the production testing phase at the foundry, an on-board processor can perform calibration and testing procedures to detect failed products. Standalone testing is possible via a loopback scheme where the processor generates test signals that are output through the DAC and are fed back into other RF/AMS components. Standalone testing facilitates qualification of SoCs in parallel, thereby reducing test time and costs, and also reducing or eliminating dependence on expensive ATE equipment that is specialized for RF/AMS validation. A processor provides all of these benefits and allows for firmware upgrades to continuously improve test coverage. Also, since the processor resides on the SoC, it has access to RF/AMS components that external equipment cannot reach due to available I/O connectivity and timing delays.

In addition to qualification testing, periodic calibration and testing of components can be conducted in the field when the device is idle or even during operation through background testing [11]. Digital correction and calibration of the RF/AMS components in the field would improve overall system performance, extend the life of the product as CMOS components slowly deteriorate, and support dynamic fault detection to meet functional safety requirements.

Additionally, advanced tools are necessary for designing and testing RF/AMS components. Cadence Virtuoso and Incisive tools provide comprehensive design and verification solutions that meet the needs of circuit and verification engineers and solve their challenges. Those solutions were applied for testing the pipeline ADC and verified its performance improvement due to the contribution of the Tensilica processor.

Additional Information

For more information on the unique abilities and features of the Cadence Tensilica Xtensa processors, see ip.cadence.com.
References


[10] “Incisive Debug Analyzer User Guide”, Product Version 14.1 June 2014. http://support.cadence.com/wps/myportal/cos/1ut/p/a0/PcmxCoMwEiDhbZ-nqahQ6NWOQQXLOU4Dhbu7UtPGrxqYidP-PZ16vZ_GpWo5oZvu4D2QUGQi29E5M53rzi24Bz1fzALD1aOjGtiWUQoJATB83g0-1r2z-fhiGg8cAESvZuquffdSzD9o73WbdvKlwlIeE9Zk8qbt31Bx-zA7ZSU1/#1030272


Solve RF/AMS Design Challenges Using the Cadence Virtuoso Solution and Tensilica DSPs