Agilent EEsof EDA

Overview on Co-Simulation of DSP, RF, Analog

Agilent Technologies

Bringing Distant Domains Under One Roof
Co-Simulation of DSP, RF & Analog

Kal Kalbsi, Andy Howard, Jack Sifri
Hewlett-Packard Company
HP EEsof Division

5601 Lindero Canyon Rd.
Westlake Village, CA 91362
818-879-6200
www.hp.com/go/hpeesof

Introduction

Modern communication products employ a multitude of technologies to cut cost, upgrade performance, speed market introduction and reduce size and manufacturability. Engineers working in the wireless industry face a particularly difficult challenge due to the growing integration of DSP, RF and analog technologies. The RF/analog engineer is asked to learn the power and flexibility of DSP and signal processing algorithms, while the DSP engineer is forced to appreciate the complexity of the RF design. The engineer/designer not only has to understand the fundamentals of each area; he needs to make correct judgements on how these domains interact. Added to this challenge has been the lack of tools capable of predicting the correct behavior of the design in an efficient and usable manner. The ability to simulate DSP concurrently with RF/analog circuitry is a powerful advantage for any designer dealing with the interaction of DSP, RF and analog technologies.

A design environment supporting a mix of simulation engines, signals and models for DSP, RF and analog technologies provides great value for top-down system specification and bottom-up test and validation (Figure 1). In this environment, simulation engines interchange data using a wide range of DSP, signal processing, RF and analog models from physical to fully behavioral. In this paper, after outlining such architecture, specific examples focusing on the mixed signal path in the modern communication system are presented.

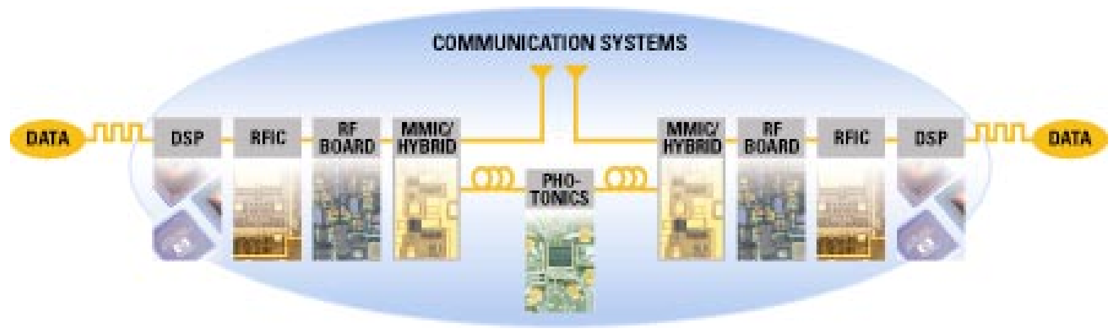

Figure 1: Communication system signal path includes DSP, photonic, RF and analog technologies.

Design flow process

Top-down flow

A successful design process includes top-down specification where broad system requirements are defined. In this process, the major blocks are initially identified and their key parameters are stated along with input/output characteristics. Next, by merging these major blocks, the interfaces are revisited and key design tradeoffs are made before finalizing what types of actual components or circuits are going to be used. During top-down design flow, the user can prototype concepts without any need for building the actual design or manufacturing a circuit. The outcome of this top-down process is a more accurate specification with well-defined building blocks and interfaces.

Bottom-up flow

A design that functions correctly as a breadboard could fail in meeting the requirements as a prototype. Unlike digital design, analog and RF designs are very susceptible to factors such as packaging, proximity and layout imprecision. These factors may cause a design to fail in meeting the specifications. In the bottom-up analysis, the designer is interested in the performance of a device and/or component in the overall system. He wants to know whether his design meets or exceeds the requirements when interacting with other parts. An RF design engineer would be interested in the impact of his design nonlinearities on the DSP algorithms while an analog designer must ensure that his design would not drift due to age, temperature variation or component tolerances. A device may be part of a component while the latter is a piece of a subsystem or system. The designer at each stage needs to know the performance metrics of the particular stage in the higher-up level.

Old Way

Regardless of whether an engineer specializes in DSP, RF or analog, designing is not a trivial task wherever these domains interact. Without co-simulation, engineers are forced to analyze the analog portion with an analog tool, the RF section with an RF tool and the DSP part with a DSP tool. Then the results must be ported from one tool to another manually and the process continued to obtain the desired outcome. The problem becomes more difficult when the analog, RF or DSP design needs to be modified or changed. This involves restarting the process from beginning to end over and over again. The process becomes a nearly impossible task, especially when the overall design includes feedback loops. If the time usually spent in porting the results of one tool to another is circumvented—a costly risk, given the time-to-market push—the designer has to put up with inaccuracies and possible wrong results as well.

New Way

Instead of struggling with porting and data format conversion from one tool to another, the designer should be able to use an environment that is designed to handle the protocols among various engines, signals and models associated with DSP, RF and analog technologies. The designer would create a design for a given technology such as RF or analog, encapsulate that design at the system level along with the rest of high-level system blocks, and simulate. The rest should be taken care of by the tool.

What is Co-Simulation?

An environment that allows a mix of circuit engines (RF, analog) with their corresponding signals and models to co-simulate with one or more data flow engines and related signals is a co-simulation environment. Such an environment allows subcircuits, which may include many analog, RF and DSP models, to simulate concurrently, exchanging their native while remaining transparent to the user.

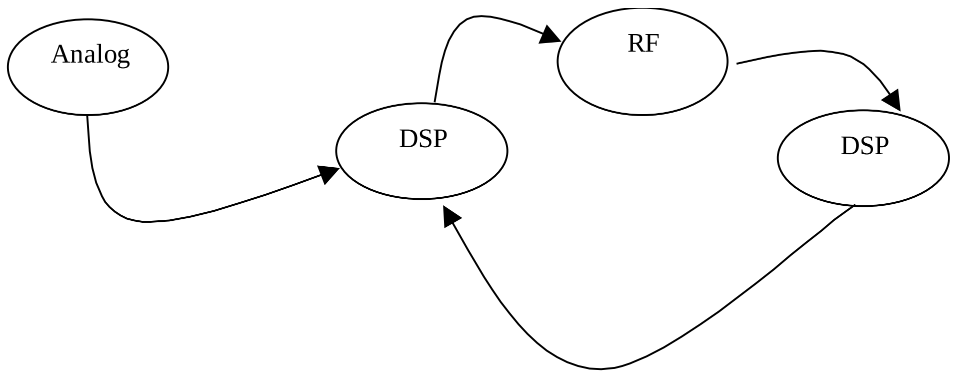

Figure 2: Subcircuits representing analog, RF and DSP designs can be connected in any topology.

Simulation, Signal & Model Space

A typical simulation tool has three components: simulation engine, certain signal representation and a collection of models. A co-simulation environment is by definition a collection of engines, signals and models. Naturally, a hierarchy of the simulators or a backplane is needed to monitor the execution of various engines. Furthermore, signals are interfaced and hence, appropriate converters are needed to transform the signals when required. There are also issues with synchronization and sampling granularity when timed signals are used. Additionally, models related to certain simulation engines should be grouped together and be differentiated during the simulation setup.

Engines

At the minimum, co-simulation of analog, RF and DSP requires at least one engine dealing with each specific technology. Each simulator or engine is based on a certain model of computation. For instance, a simulator may be a solver of a (partial) differential or integral equation imposing certain global constraints on the problem boundary in space or time. Circuit simulators are designed to solve the Kirchhoff's voltage or current law, in frequency- or time-domain. There exist various models of computation such as convolution, harmonic balance, or a combination of these methods, which are used by analog and RF simulators.

In the analog/RF area, a SPICE-type simulator handling the transient effects is a must. Because SPICE simulators cannot handle RF signals efficiently, a time-domain circuit simulator for the RF signals is needed as well. Unlike SPICE, the RF simulator samples the baseband complex envelope of the signal instead of its RF carrier. In some advanced RF circuit tools the RF carrier is simultaneously computed in the frequency-domain for each envelope time sample.

In the signal-processing realm the appropriate engine can be based on data flow semantics. Data flow technology in its various forms (synchronous, Boolean, dynamic, etc.) is ideal at the behavioral abstraction level. In a data flow simulation, each element fires only when sufficient data is available for it to consume. A schedule determines the sequence of firings based on the availability of data. The engine utilizes numeric-type signals to handle the DSP and signal processing portions of the signal path.

Signals

Each engine addresses one or more signal types. Signals are the language simulators use to communicate. A signal can be perceived as a collection of data and attributes versus an independent variable. Examples are time waveform with floating point data, time- or multi-dimensional data, and numeric counter or index. A signal could be physical (time or frequency signals) or completely non-physical.

For DSP algorithm development, the engineer needs access to a broad range of signals, including integers, floating points, and complex- and fixed-point numbers, both as scalars, vectors or matrices. These signals are used for algorithmic development in the baseband portion of a communication system. Beyond the algorithmic development, and between the D/A and A/D converters, the designer needs time-domain signals. In addition, since the analog and RF simulators deal with time-domain signals, converters are needed to transform the numeric signals to time-domain waveforms and vice versa.

Models

Both circuit and data flow simulators use a variety of models. These models have parameters reflecting the physical dimensions and properties of a device or component. There are also behavioral models that are purely mathematical or abstract in nature.

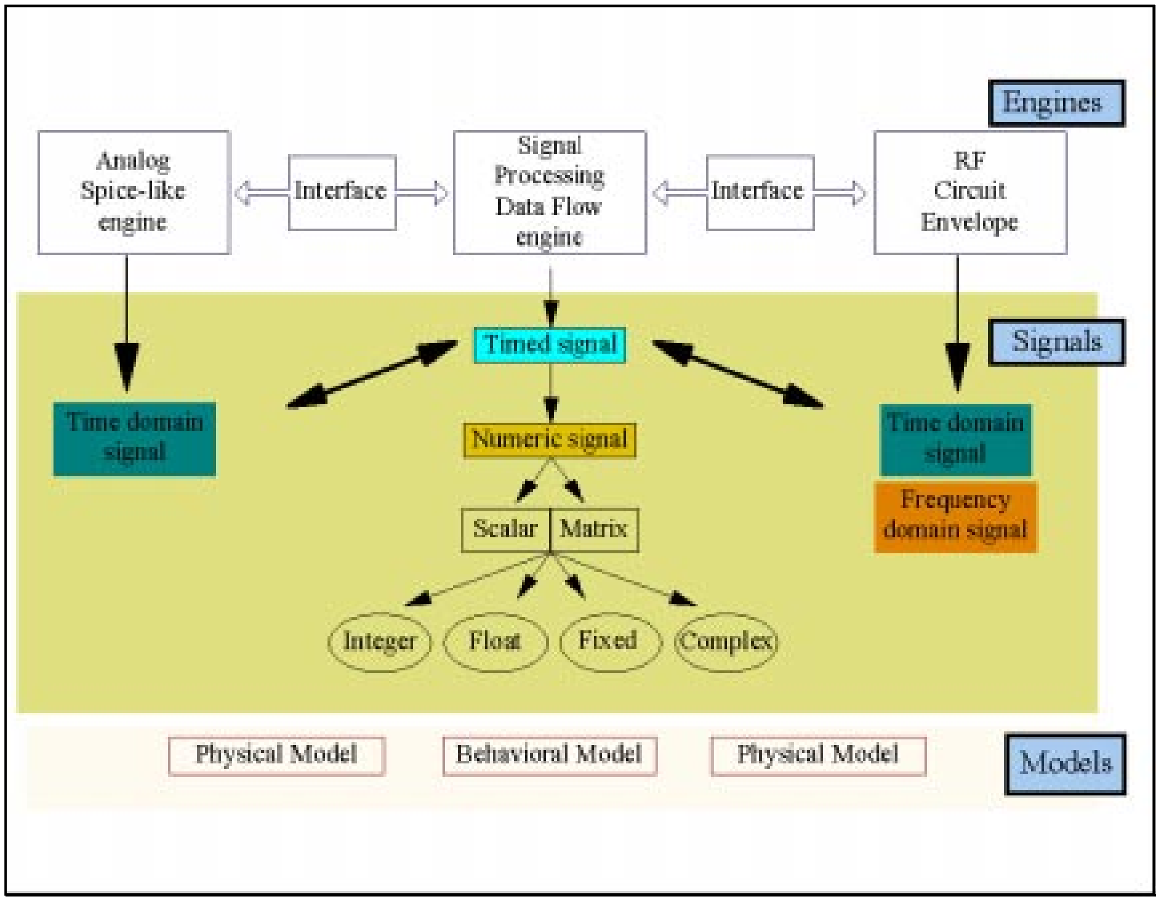

Diagram 1: The co-simulation environment for DSP, analog and RF is comprised of engines, signals and models:

How does it work?

In the co-simulation environment described above, circuit designs are first created on the circuit schematic. This circuit design can be tested using appropriate circuit sources and measurements with either the RF or analog simulators. Once the circuit design has been verified, ports to be interfaced with the signal processing design are identified and placed. Next, an instance of the design is placed on the signal processing schematic and connected to the other blocks. The combined schematic design can now be simulated.

On the signal processing page, a circuit subnetwork is just a component with a certain number of input and output ports. This circuit subnetwork is part of the schedule determined by the data flow engine. It would be fired just like any other component according to the schedule, and as many times as required. Every time the circuit subnetwork is fired, the circuit simulator continues to carry on the simulation based on the input it receives from the signal processing interface. Once the circuit simulator is finished with its analysis, it passes the simulation results back to the signal processing interface. This cycle repeats as many times as the scheduler requires. The duration of the circuit simulation each time it is invoked is determined by the time step provided by the connecting signal processing component.

From the circuit simulator viewpoint, the signal processing input interface is deemed an ideal source. The more ports at the input interface to the circuit, the more ideal sources there will be feeding the circuit subnetwork. At the output interface of the circuit, the results are shared with the connecting signal processing component.

Clustering is the process of defining the boundaries of the signal processing, analog and RF simulators. Initially, this boundary is defined by circuit schematics, where the analog or RF subnetworks are defined and then an instance of those are made on the signal processing schematic.

At each interface boundary between signal processing, analog and RF simulators, there needs to be an exchange of information. The semantics and fundamentals of simulation in each application area are quite different. Time samples for signal processing are one fixed time step apart. However, both RF and SPICE simulators refine the time according to the various internal criteria. The common signal being exchanged between signal processing data flow components and the circuit simulators is a time-domain signal. Hence, all three engines deal with the notion of time step. The signal entering the circuit subnetwork should be aware of "time."

Applications

Fractional Divider

The fractional-divider phase-locked loop is a good example of the tight integration of DSP and RF/analog on a key component (i.e., PLL) in a communication system. This example shows how co-simulation of DSP and analog engines can help the engineer conceptualize and verify a mixed DSP/analog design before resorting to a prototype. The example includes a feedback path between the DSP and analog part, showing the value of co-simulation as opposed to using isolated tools. In this example, the PLL is created on the circuit page, and the fractional synthesis part is created on the signal processing page. Time-domain waveforms are used for PLL simulation, while fixed-point signals with appropriate bitwidth and precision are used for the fractional part. Appropriate converters between the two signal types are used when needed.

In a conventional PLL, $f_{vco} = N * f_{ref}$ where f_{vco} is the VCO frequency, f_{ref} is the reference or loop frequency and N is the integer divide ratio. In a fractional-N synthesizer, the divide ratio is periodically toggled between N and $N+\delta N$, which allows a finer loop frequency. Consequently, the average divide ratio will be increased from N to $N.f$, where $.f$ is the fractional part of the average divide ratio and the new VCO frequency relation is $f_{vco} = N.f * f_{ref}$. Typically, the overflow from an accumulator is used to modulate the instantaneous divide ratio.

The drawback of this new approach is the fractional spurs at all multiples of the offset frequency ($.f * f_{ref}$). Using a correction scheme called phase interpolation, fractional spurs in the order of –70 dBc have been achieved, but the complexity and expense of interpolation circuitry makes this approach undesirable.

Using a similar approach such as the ones described in references [2] and [3], the fractional synthesis is combined with sigma-delta modulation to achieve frequency synthesis with spurious-free output and near-infinite resolution. This approach has been successfully simulated via co-simulation of DSP and analog engines and models.

The analog part of the design is shown in Figure 3, where a modified version of the DECT (digital European cordless telephone) PLL is assembled on the circuit schematic page. The modification has to do with a 10X increase in loop bandwidth for better visualization of results. This PLL design includes, among other parts, a phase-frequency detector, a loop-lumped element filter, and a VCO model with the divide by N part integrated in the model. This analog design was initially simulated stand-alone, with results identical to a conventional PLL.

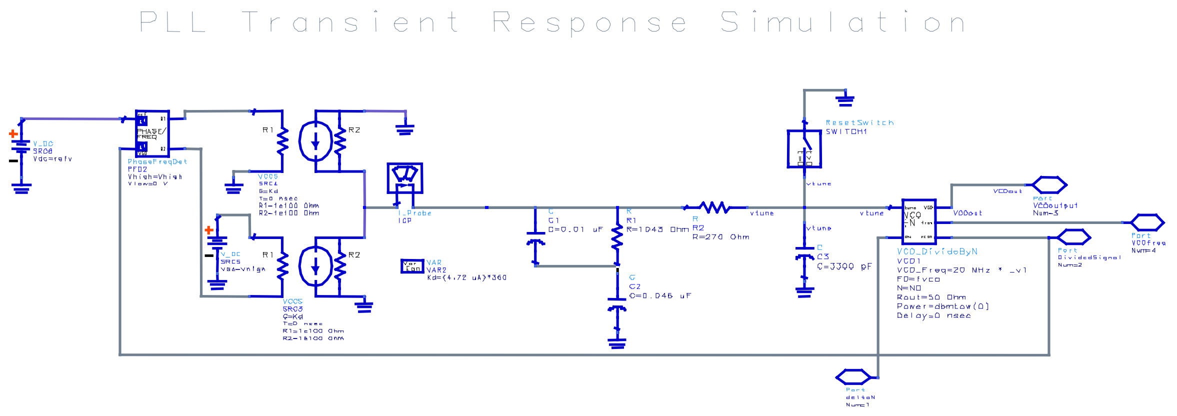

Figure 3: PLL design using an analog engine and time-domain signal.

The top-level design in Figure 4 shows the encapsulation of the analog PLL and the DSP portion, which includes the multiple stage modulator divider implemented with accumulators. The desired fraction, such that $f_{vco} = N.f * f_{ref}$, is set at the input to the multiple modulator divider. Using multiple stages of a synthesizable adder and data register models with the output precision set to

10 bits, the accumulator overflows at the value (*N=2**10-1=1023*). Given a f_{ref}=1.728 MHz, and the nominal divide ratio of 1023, f_{vco} = 1.767744 GHz, but because δN is changing with time, f_{vco} also changes with time. The simulation was set up so that after 300 samples, the input to the first adder changes from 0 to 500. The simulation results indicate approximately how long it takes for the VCO to transition from $N * f_{ref}$ to $N.f * f_{ref}$. Figures 5 and 6 depict the fractional divide ratio change and frequency error vs. time

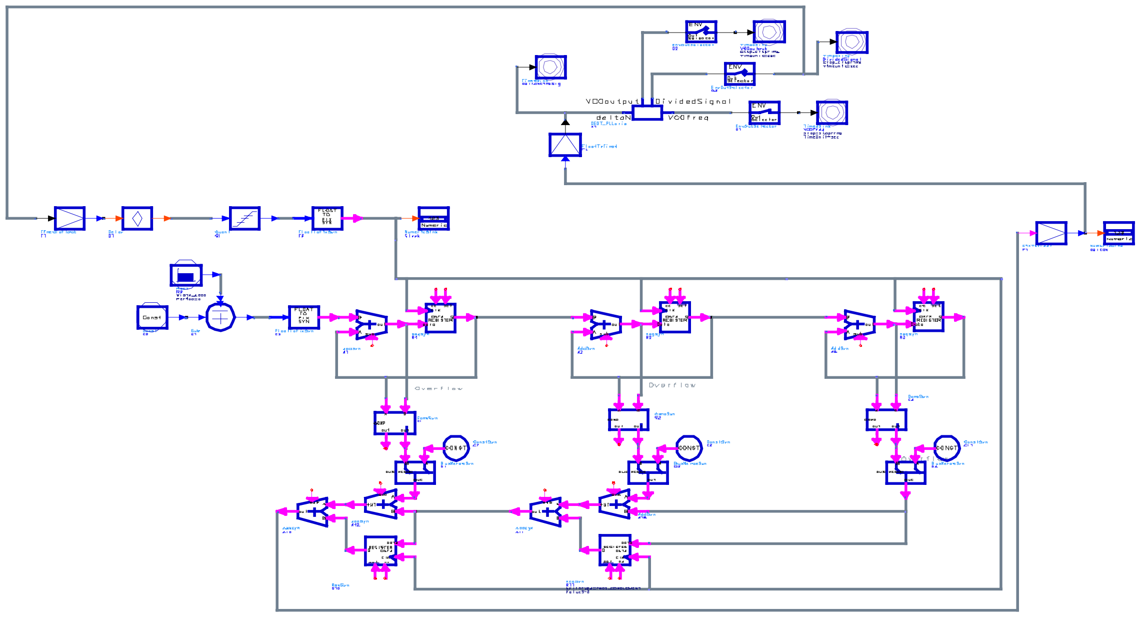

Figure 4: Top-level design with PLL encapsulation and the DSP part using data flow engine and fixed point signal.

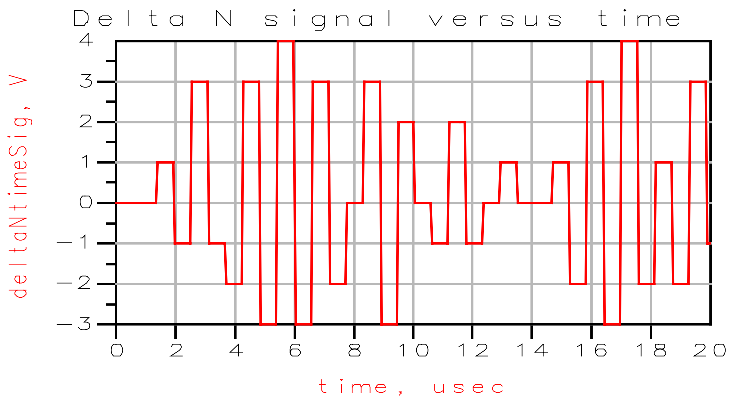

Figure 5: Fractional increment deltaN versus time.

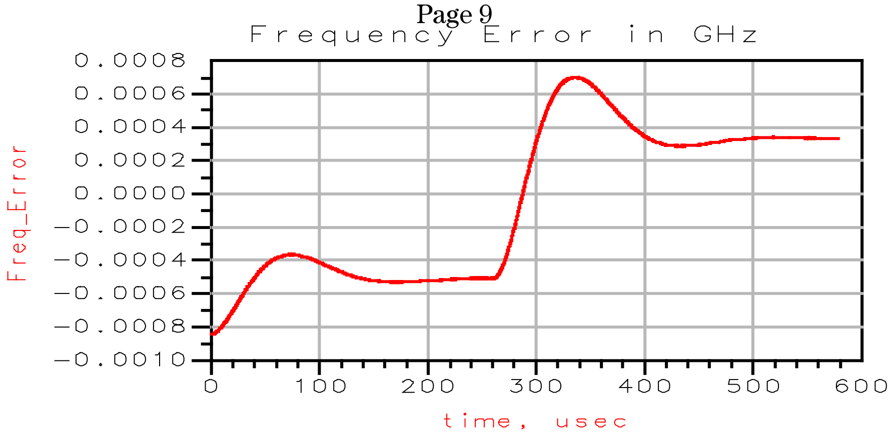

Figure 6: Frequency error versus time.

RFIC example

Figure 7 represents an example of how a mix of analog/RF and dedicated on-chip DSP blocks can be simulated, designed, and verified using the co-simulation environment. The DSP 16 QAM I Q data generator chip has been designed and analyzed using the data flow engine, while the RFIC has been designed using the RF simulator along with SPICE-like simulator.

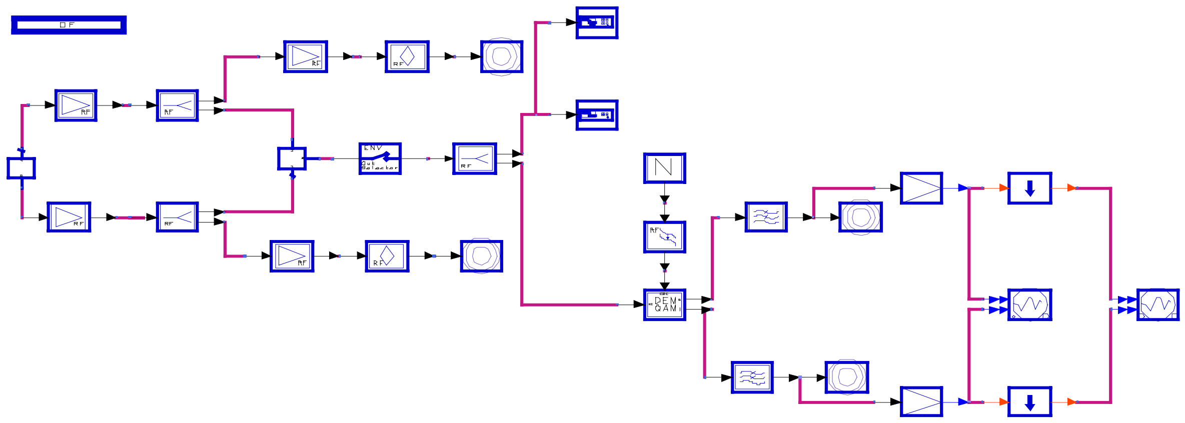

Figure 7

Figure 8 is the DSP portion of the design showing the bit source stream into serial to parallel; upsampler and filter blocks are all in fixed point signal type. In addition, the design includes control and reset blocks. The output of the filter is then converted to “analog” form by the appropriate converter.

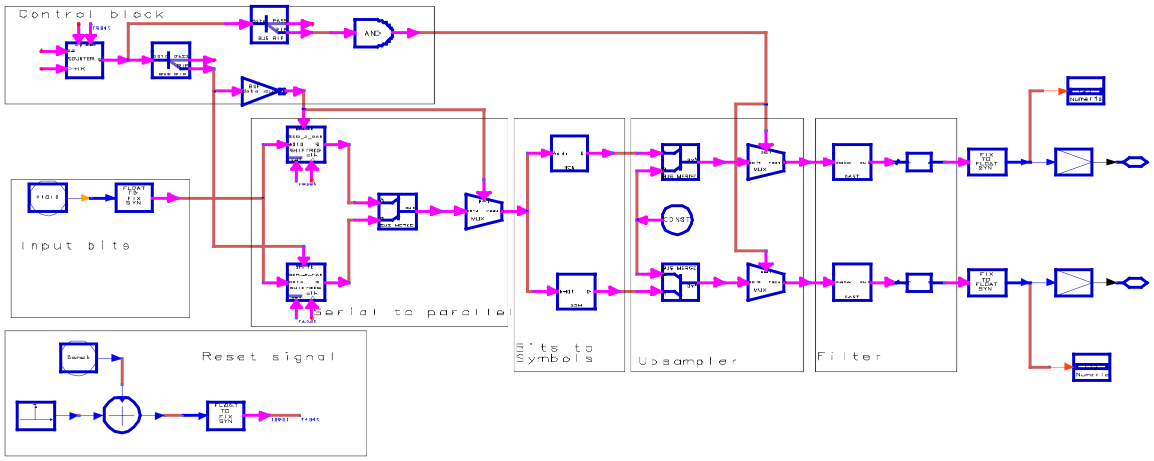

Figure 8

The bit-to-symbol section includes 4-bit bytes for I and Q channels (in practice these would be merged into one ROM). The symbols are then upsampled with a rate of 4. The root raised cosine filters were designed using a filter design tool with a rectangular window with a bit width of 13, resulting in a 90 taps design with a suppression of 40 dB in the stop band.

The I and Q outputs from the filters need to be upconverted to a frequency of 5 MHz. For a sampling rate of 20 MHz, this implies that there are 4 samples per cycle of the 5 MHz sine and cosine waveforms. Using 1, 0, -1 and 0 results in the output of the upconverter as indicated.

cos( )	sin( )	I cos( ) - Q sin( )
1	0	I
0	1	-Q
-1	0	-I
0	-1	Q

Table 1

Thus, to up-convert the I and Q signals to 5 MHz carrier, the I and Q signals merely have to output in the order I, -Q, -I and Q. This is done using a MUX. The system-level amplifier blocks here are used only to amplify or attenuate the signal level going into the modulator.

The IQ modulator consists of an LO source at 2 GHz driving two Gilbert cells. Using a Wilkinson combiner, the outputs of the Gilbert cell mixers are combined and amplified using a 2 GHz double-stage power FET amplifier.

The circuit implementation of the Gilbert cell mixer with the bipolar transistors is shown in Figure 9. The 2 GHz two-stage output uses NEC FETs. FETs are used here along with microstrip transmission lines and bipolar devices in order to demonstrate the diverse use of components and elements in the mixed-signal co-simulation environment.

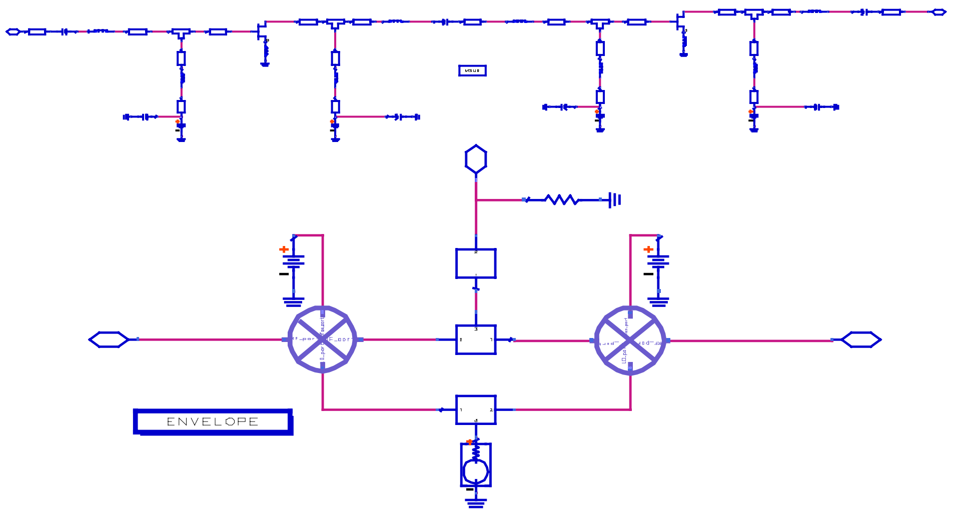

Figure 9

In order to compare the input I-Q data to the output received data, the output spectrum was demodulated back, recovering I and Q data and comparing it to the original data with reasonable correlation, as shown superimposed in Figure 10. If the output amplifier is removed or its gain is reduced, the constellation becomes much closer to the nominal values with improved error vector magnitude and bit error rate.

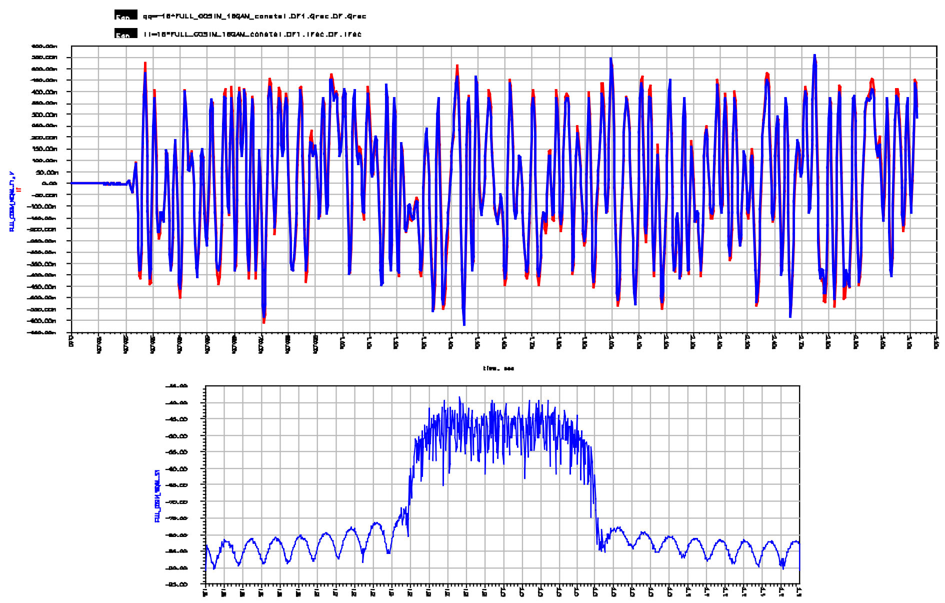

Figure 10

Summary

Simulation of behavioral DSP designs along with analog/RF circuit designs is critical to the success of the integrated components, devices, and subsystems used in modern, wireless applications. It is essential to verify the impact of real-world analog/RF issues on DSP algorithms and vice versa in a highly integrated environment.

For designs of low complexity, it is possible to use separate simulators for the signal processing and analog/RF portions and then integrate the results. However, today's state-of-the-art designs using a mix of analog/RF and dedicated on-chip DSP blocks require high levels of integration at the two-environment boundary. The co-simulation between signal processing and circuits offered by HP Advanced Design System addresses this need.

References

[1] Advanced Design System, Hewlett-Packard Company, HP EEsof Division

[2] Brian Miller and Robert Conley, “A Multiple Modulator Fractional Divider" *IEEE Transactions on Instrumentation and Measurement*, June 1991.

[3] "Technique Enhances the Performance of PLL Synthesizers" *Microwaves & RF* January 1993.

For more information about Agilent EEsof EDA, visit:

www.agilent.com/find/eesof

Agilent Email Updates

www.agilent.com/find/emailupdates
Get the latest information on the products and applications you select.

Agilent Direct

www.agilent.com/find/agilentdirect
Quickly choose and use your test equipment solutions with confidence.

www.agilent.com

For more information on Agilent Technologies' products, applications or services, please contact your local Agilent office. The complete list is available at:

www.agilent.com/find/contactus

Americas

Canada	(877) 894-4414
Latin America	305 269 7500
United States	(800) 829-4444

Asia Pacific

Australia	1 800 629 485
China	800 810 0189
Hong Kong	800 938 693
India	1 800 112 929
Japan	0120 (421) 345
Korea	080 769 0800
Malaysia	1 800 888 848
Singapore	1 800 375 8100
Taiwan	0800 047 866
Thailand	1 800 226 008

Europe & Middle East

Austria	0820 87 44 11
Belgium	32 (0) 2 404 93 40
Denmark	45 70 13 15 15
Finland	358 (0) 10 855 2100
France	0825 010 700*
	*0.125 €/minute
Germany	01805 24 6333**
	**0.14 €/minute
Ireland	1890 924 204
Israel	972-3-9288-504/544
Italy	39 02 92 60 8484
Netherlands	31 (0) 20 547 2111
Spain	34 (91) 631 3300
Sweden	0200-88 22 55
Switzerland	0800 80 53 53
United Kingdom	44 (0) 118 9276201

Other European Countries:
www.agilent.com/find/contactus

Revised: March 27, 2008

Product specifications and descriptions in this document subject to change without notice.

Printed in USA, January 01, 1999
5989-9004EN

Agilent Technologies