Presented at the Thirty-Second Annual Asilomar Conference on Signals, Systems, and Computers - November 1998

Cosimulating Synchronous DSP Applications with Analog RF Circuits

José Luis Pino and Khalil Kalbasi

{jpino,ekalbasi}@wlv.hp.com

Hewlett Packard, EEsof Division

Abstract

This paper introduces timed synchronous dataflow (TSDF) which enables the codesign of the synchronous DSP and analog RF portions of an application. The semantics and scheduling techniques of TSDF are detailed. A 16 QAM modem with a QAM synthesizable DSP transmitter, cosimulating with a RF modulator and RF power amplifier is demonstrated.

1 Motivation

Many of today's applications demand a tight integration of analog RF and DSP technologies. This can be readily seen in applications such as television and cellular telephony. Both of these applications have traditionally been implemented solely with analog, but as DSPs have become faster, smaller and more power efficient new standards have been adopted to make use of DSP. The integration of analog RF and DSP is likely to quicken as systems on a chip become practical with the advent of silicon germanium and silicon-on-insulator-based CMOS.

The traditional design flow unfortunately is inadequate to tackle the new level of integration between analog RF and DSP. The design flow typically begins with a small group of experts partitioning the design into the analog and DSP portions. Once the application is partitioned and the interfaces defined, separate teams design the analog RF and DSP portions. The design typically is conducted with very little interaction between the two groups. When the designs are complete a prototype is constructed in what is known as the integration and test phase. This phase can take up to 90% of the design time for the application. We believe the time needed in the last phase of the design can be greatly reduced by enabling the codesign of the analog RF and DSP portions.

Because of the traditionally disjoint design of analog RF and DSP systems, most EDA tools have evolved to target one of these two areas. However, to foster the codesign of analog RF and DSP, EDA tools must allow the analog RF and DSP portions to cosimulate throughout the design cycle.

In this paper, we introduce timed synchronous dataflow (TSDF) which provides a backplane on which synchronous DSP portions can be cosimulated with the analog RF portions of the design.

The structure of the paper is as follows. In the first section, we review synchronous dataflow, transient, and circuit envelope simulation technologies. Using these simulators, a designer can specify the various elements in a mixed analog RF and DSP application. Next, we construct TSDF which enables the cosimulation of all three simulation technologies. Finally, we detail a 16 QAM transmitter, complete with a synthesizable DSP section, a RF modulator and a two stage power amplifier.

2 Background

In this section, synchronous dataflow (SDF), SPICE and circuit envelope simulation technologies are reviewed. Each of these simulation technologies is ideal to design and simulate a portion of the mixed analog RF and synchronous DSP applications. Table 1 summarizes each of the simulation technologies.

2.1 Synchronous Dataflow

Dataflow is a natural representation for signal processing algorithms. One of its strengths is that it exposes parallelism by expressing only the actual data

Simulation Technology	Application
SDF	Synchronous DSP, dataflow simulation
SPICE	Analog baseband, time domain simulation
Circuit Envelope	Analog RF simulation, hybrid frequency domain/time domain simulation
TSDF	Behavioral analog RF, timed dataflow simulation

Table 1: Summary of simulation technologies

dependencies that exist in an algorithm. Applications are specified by a dataflow graph in which the actors represent computations, and data tokens flow between them along the arcs of the graph. HP Ptolemy [1,4] is a framework that supports dataflow programming.

There are several forms of dataflow defined in HP Ptolemy. Synchronous dataflow(SDF), allows the succinct specification of synchronous DSP applications. A synchronous DSP application is one in which all of the sampling rates in the system are rationally related. SDF is ideally suited to a large set of DSP applications such as digital communication (QAM, PSK, CDMA) and filtering applications (wavelets, filterbanks, IIR, FIR).

In SDF [7], the number of tokens produced or consumed in one firing of an actor is constant. This property makes it possible to determine execution order and memory requirements at compile time. Thus these systems do not have the overhead of run-time scheduling, and have very predictable run-time behavior.

Figure 1 shows a simple SDF graph. In this graph, actor *A* produces two tokens and actor *B* consumes three tokens for each firing. In a *periodic* SDF schedule, the first-in/first-out (FIFO) buffer on each arc returns to its initial state after one schedule period. For each actor x in a properly constructed SDF graph, there exists a positive integer $\mathbf{q}(x)$ such that actor x must be invoked at least $\mathbf{q}(x)$ times in each period of a periodic schedule [7]. For the example in figure 1, $\mathbf{q}(A) = 3$ and $\mathbf{q}(B) = 2$. The vector containing all of the solution for $\mathbf{q}$ for each actor is known as the repetitions vector.

Given an SDF specification, we can construct a periodic schedule at compile-time that can be iterated an indefinite number of times without requiring unbounded memory. Such a schedule can be constructed by invoking each actor x exactly $\mathbf{q}(x)$ times, and ensuring that the data precedences defined by the SDF graph are respected. For figure 1, one such schedule is $AABAB$. In general there are can be many schedules for an SDF graph.

$$\begin{tikzcd} A \arrow[r, "2\quad\quad 3"] & B \end{tikzcd}$$

Figure 1: A simple SDF graph.

2.2 SPICE

SPICE simulation has existed for many years and has been used to design baseband analog circuits. It provides a framework on which analog nonlinearities and distortions can be modeled.

SPICE is an analog time domain simulator and as such relies on sampling the signals. To simulate an analog circuit, the simulator constructs a system of differential equations for each node. These differential equations represent the Kirchhoff's current law (KCL) requirements. The simulator then solves these equations iteratively using the Newton Raphson algorithm.

Although SPICE can be used to simulate RF circuits, it is too expensive. The reason for this is that time step is upper bounded by the RF signal and the simulation time is lower bounded by the period of the lowest frequency in the simulation. The lowest frequency in the simulation is typically the spacing between two RF carrier frequencies [6].

2.3 Circuit Envelope

Circuit envelope[3] addresses this deficiency in SPICE by allowing a circuit to be simulated using a hybrid time domain and frequency domain engine. During simulation, each signal is represented as time varying spectra. Figure 2 shows how the simulator represents a signal. By separating the carriers from the envelope, the sampling requirements are reduced to that required to represent the envelopes. Because of this separation of carrier and envelope data, circuit envelope is ideal to model analog RF circuits.

3 Timed Synchronous Dataflow

TSDF extends SDF by adding four concepts to SDF. First, TSDF adds a timed data type that can represent a

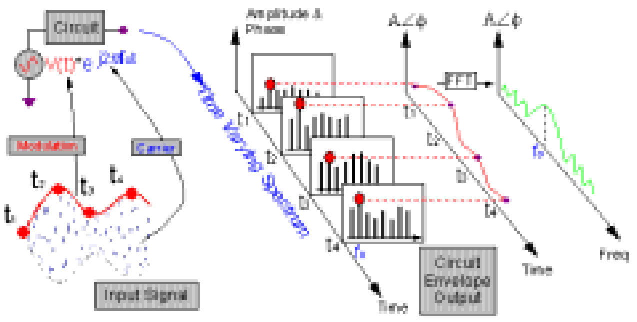

Figure 2: Circuit Envelope

signal as an envelope and carrier frequency (f_c). Second, TSDF makes a distinction between timed and numeric actors. Third, TSDF input and output pins can define an optional impedance. Lastly, TSDF defines that the timed data tokens produced from a timed actor are equally spaced in time.

A timed actor is a TSDF actor which maintains a concept of time and obeys SDF semantics. A TSDF actor is defined to fire at constant rate. In any given simulation, all timed actors are executing simultaneously. This is akin to the concurrent execution of the individual components in an analog circuit. Likewise, the timed data tokens produced by a timed actor are defined to be equally spaced. The time step between samples for each arc are computed at compile time, by a time step propagation algorithm described in section 3.1.

The timed data signal representation is similar to the signal representation in circuit envelope. A timed data token is a made up of four elements:

- I(t) - amplitude of the in-phase component of envelope
- Q(t) - amplitude of the quadrature-phase component of the envelope
- t - current time
- f_c - carrier frequency of the signal

The value of the signal is:

$$\mathrm{I}(t)\cos(2\Pi f_c t) - \mathrm{Q}(t)\sin(2\Pi f_c t)\,.$$

Thus by the extending SDF with time, TSDF provides a mechanism to describe functional models of analog RF components. Examples of TSDF components include timed FIR filters, and antenna/propagation models[5].

3.1 Time Step Propagation

Any timed actor in a TSDF graph may optionally define a sampling rate for any of its pins that produce or consume timed data. In the simulation initialization phase, the sampling rate is propagated over all arcs in the graph. If more than one pin specifies a sampling rate, then the sample rates propagated must be consistent. If an inconsistency is found, the graph is declared to be incorrectly specified.

The time step propagation algorithm is broken down into two procedures, one an initialization phase; the second a recursive procedure which propagates the time step.

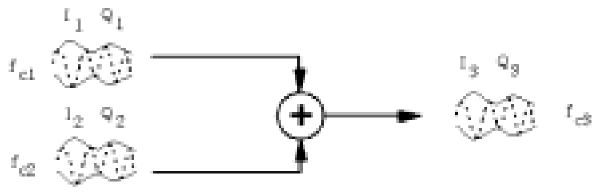

Figure 3: RF Signal addition in TSDF

The algorithm is shown below:

1. Compute the repetitions vector for the TSDF system (see section 2.1)
2. For each connect graph G in the TSDF specification find an arc with the time step > 0. If no such arc exists in graph G, continue onto the next graph.
3. For the arc found in step 2., compute the time step of the source actor by multiplying the SDF parameter of the source pin by the arc time step. Call the procedure `propagateTimeStep` with the source actor, S, and the time step, T, of the source actor as arguments.
4. Repeat step 3 for the sink pin.
5. Continue onto the next graph.

The `propagateTimeStep` recursive function is shown below.

1. For each pin of the actor S, compute the time step by dividing the time step T by the SDF parameter of the pin.
2. If the pin time step > 0, compare it with the value calculated in step 1. If the value is not the same, declare the graph as inconsistent and stop.
3. If the pin time step = 0, set the pin time step of the arc. Compute the time step of actor at the far end of the arc. Call `propagateTimeStep` with new actor and its time step.

3.2 Carrier Frequency Propagation

As with the time step propagation algorithm, the value of carrier frequencies are determined in the simulation initialization phase. Unlike the time step propagation algorithm, there is no closed form solution and thus heuristics must be used. The carrier frequency values are used to convert between timed and other data types as well as by the timed components.

An example of how a timed component can use the carrier frequency is the RF signal adder shown in figure 3. Here the envelopes of the two input signals are combined and new carrier frequency is computed. As can be seen by this example, the carrier frequency propagation is function dependent.

After propagation, the carrier frequency for each arc will be either non-negative or undefined. All timed arcs have a non-negative carrier frequency; all other arcs have an undefined carrier frequency.

The carrier frequency propagation algorithm is listed below:

1. Calculate the topological sort for the graph. To accommodate feedback paths, the classical topological sort algorithm is modified by specifying an input ordering of the actors for the initial depth first search. To construct the input ordering, list all of the source actors at the beginning of the list and append the remaining actors.

2. Iterate over the sorted list of actors, allowing each actor to propagate the carrier frequency.

For feedforward designs, the carrier frequency propagation solution is unique and converges quickly. For designs with feedback, heuristics are needed to find a solution. In these designs, step 2 can not be completed because the carrier frequency of the inputs from the feedback arcs will have an unspecified carrier frequency. For these designs, we must assume an initial carrier frequency for the feedback arcs. The algorithm to set the carrier frequency is:

1. Set the unknown time step for feedback inputs to the maximum over the other inputs.
2. Propagate the carrier frequency using the sorted listed above.
3. If the carrier frequency is converged at all pins, then the carrier frequency resolution is complete.
4. If the assumed carrier frequencies failed to converge, repeat steps 1and 2 with the default carrier frequency set to 0.
5. If converged, carrier frequency resolution is complete; otherwise, it has failed and the end user will have to explicitly set the carrier frequency.

4 16 QAM Modem

In this section, we give an overview of an 16 QAM modem design that demonstrates the codesign of analog RF and DSP. This example is from a short course available from HP EEsof[2].

The top level design is shown in figure 4. It is specified

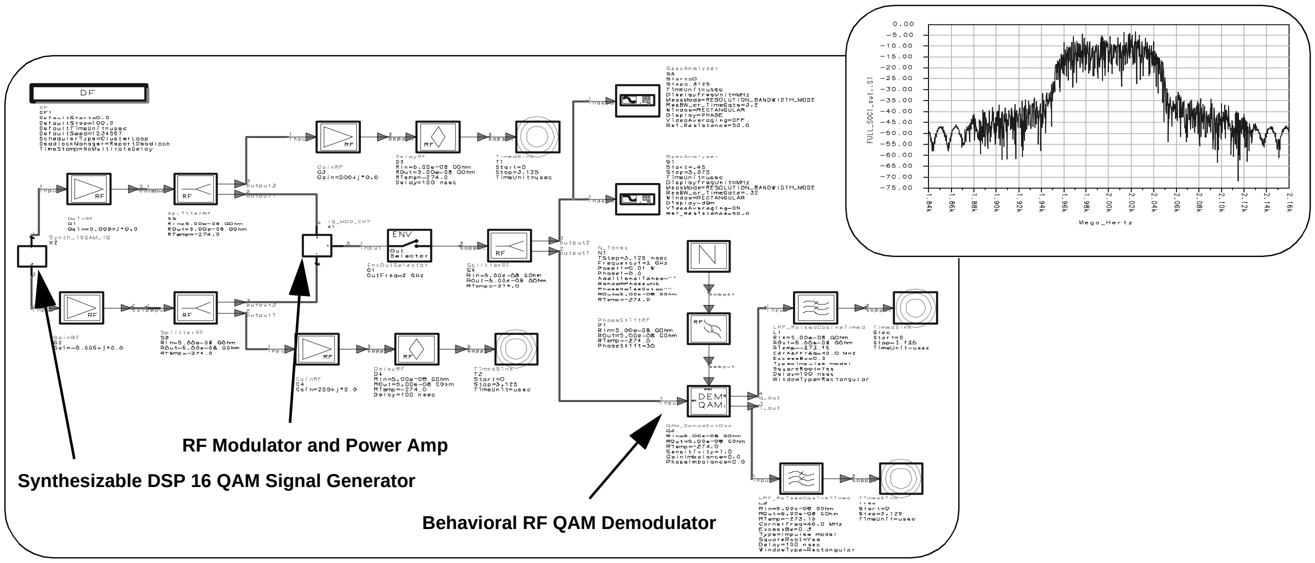
RF Modulator and Power Amp
Synthesizable DSP 16 QAM Signal Generator
Behavioral RF QAM Demodulator
Mega_Hertz

Figure 4: The top level specification of 16 QAM system and a plot of the spectrum of modulated RF signal. The DSP section is detailed in figure 5; the analog RF section is detailed in figure 6.

with TSDF, which enables the cosimulation of the SDF and circuit envelope.

Within this schematic, numeric simulation is embedded within the DSP 16 QAM signal generator shown in (figure 5). From this schematic, Verilog or VHDL can be generated.

The RF section is shown in figure 6. In this section the signal is modulated with a 2 GHz RF carrier and the modulated output is then passed through a two stage power amplifier.

5 Conclusions

Timed synchronous dataflow enables efficient cosimulation between the synchronous DSP and analog RF portions of an application. It does this by extending SDF with explicit sampling rates on each arc and maintaining a representation of RF signals as a sampled envelope with a RF carrier frequency. By enabling the codesign of the various analog RF and DSP components throughout the design cycle, various long integration and test parts of the design cycle can be reduced.

References

[1] J. Buck, S. Ha, E. A. Lee, and D. G. Messerschmitt, “Ptolemy: A framework for simulating and prototyping heterogeneous systems,” *International Journal of Computer Simulation, special issue on Simulation Software Development*, vol. 4, 1994. http://ptolemy.eecs.berkeley.edu/papers/JEurSim

[2] Co-Design/Co-Simulation Lab, Hewlett Packard, HP EEsof Division. http://www.tmo.hp.com/tmo/hpeesof

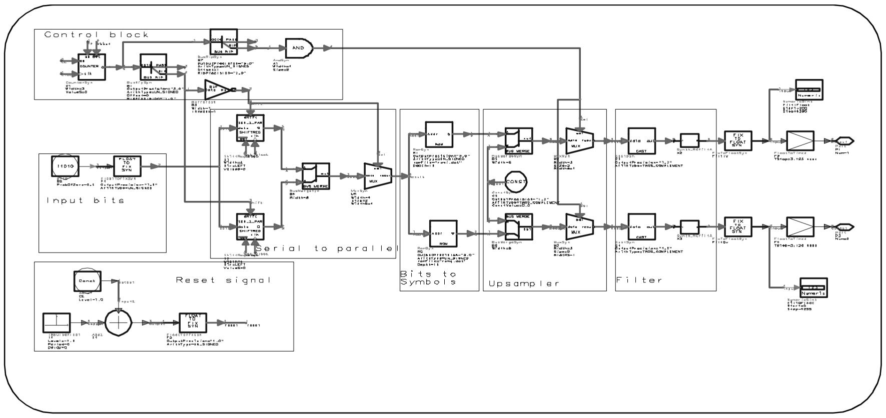

Figure 5: Synthesizable DSP 16 QAM specification.

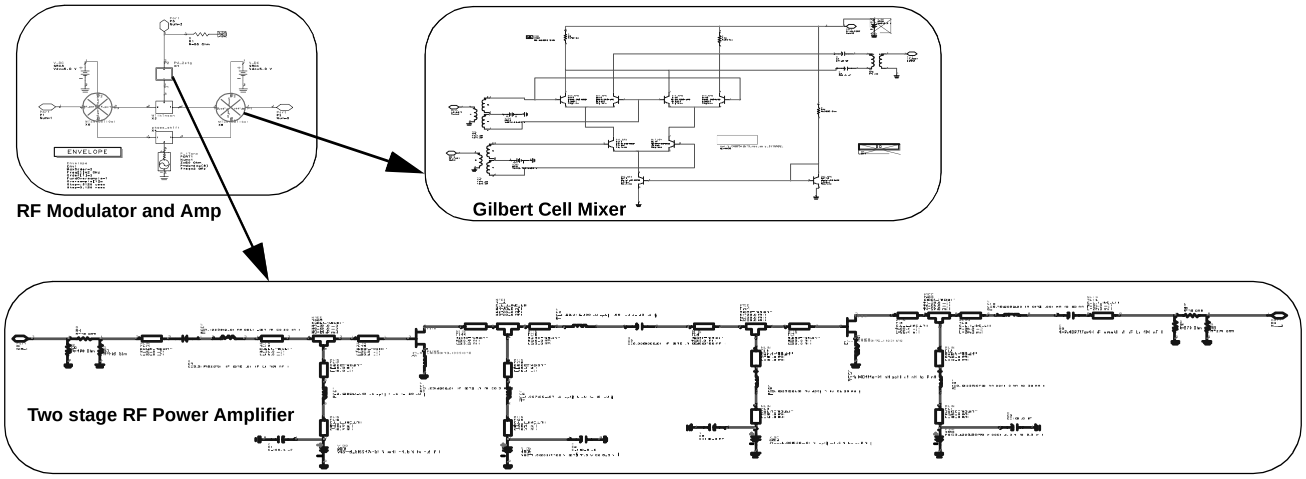

Figure 6: Top level and detailed specification of RF modulator and power amplifier sections.

[3] A. Howard, “Circuit Envelope Simulator Analyzes High-Frequency Modulated Signals,” *RF Design*, September 1995.

[4] HP Advanced Design System, Hewlett Packard, HP EEsof Division. http://www.tmo.hp.com/tmo/hpeesof

[5] K. Kalbasi, “Use Propagation Models to Simulate Interference and Diversity Scenarios,” *Wireless Systems Design*, January 1996.

[6] N. Kanaglekar and J. Sifri, “Integrate CAD Methods For Accurate RF IC Simulation”, *Microwaves & RF*, November 1997.

[7] E. A. Lee and D. G. Messerschmitt, “Synchronous data flow,” *Proceedings of the IEEE*, vol. 75, no. 9, 1987.