Pareto Optimal Modeling for Efficient PLL Optimization

Saurabh Kumar Tiwary, Senthil Velu, Rob A Rutenbar and Tamal Mukherjee

Electrical and Computer Engineering Department
Carnegie Mellon University, Pittsburgh PA USA
{stiwary, snv, rutenbar, tamal}@ece.cmu.edu

ABSTRACT

Simulation-based synthesis tools for analog circuits [1,2] face a problem extending their sizing/biasing methodology to larger block-level designs such as phase lock loops or converters: the time to fully evaluate (i.e., to fully *simulate*) each complete circuit solution candidate is prohibitive inside a numerical optimization loop. In this paper, we show how to circumvent this problem with a careful mix of behavioral models for less-critical parts of the block, and pareto-optimal *trade-off models* for the critical components. In particular, we show how to adapt current circuit synthesis techniques to build the required tradeoff models. As a concrete example of the methodology, we show detailed simulation results from the synthesis of critical portions of a 500MHz digital frequency synthesizer PLL.

Keywords: behavioral modeling, pareto-optimal design, analog circuits, phase locked-loop, circuit optimization.

1 INTRODUCTION

Recent advances in design automation and increased computational capability of computers has led to a gradual transition from 'hand-calculation' based analog circuit design to a simulation-based sizing methodology. Simulation based synthesis uses efficient global optimization to visit many circuit candidates, and fully evaluates each candidate via detailed simulation. This methodology works very well for circuits having in the range of a few hundred devices. However, for larger circuits, the simulation time required for a single simulation is too large to do a practical simulator-in-the-loop circuit sizing. Also, due to the curse of dimensionality, the design space in which to search for optimal design points becomes too large for these circuits to be handled by the tools available today. Of late, much work has been done in the field of analog circuit *macromodeling* which aims at simulating these circuits faster [3], [4], [6]. This work proposes to adapt these behavioral modeling approaches to model the non-critical blocks of a large system level design and then use *pareto trade-off curves* to search for optimal designs of critical blocks to meet the overall system level performance specifications. A phase-locked-loop has been used to as a vehicle to show the results for our suggested methodology.

In section 2, an overview of the problem and the proposed methodology is presented. Section 3 gives a brief summary of the behavioral modeling methodology. Generation of pareto trade-off curves are also discussed in this section. In section 4, the proposed method is applied to a 0.18um digital frequency synthesizer and results are presented. Finally, conclusions are presented in section 5.

2 OVERVIEW

2.1 Existing Methodologies

The simulation based circuit sizing methodology is a well researched field. The circuit designer chooses a circuit topology which is presented to the sizing tool with an approximate range in which the various transistor sizes should be. The sizing tool [7] uses global optimization techniques to search for an optimal design point which satisfies the circuit specifications as given by the designer. This approach uses a simulator "in-the-loop" to characterize each visited design point; hence, the time taken for convergence often is proportional to the time taken to complete the simulations for a single design point and the size of the design space. This approach thus, is not directly suitable for circuits with a very large number of design parameters. Also, circuits whose performance specifications require long simulation runs, would take too long a time for synthesis. The phase noise and settling time for a phase locked-loop are examples of such specifications which require a long time for evaluation for a single design point.

Researchers have looked at the problem of simulating large analog circuits efficiently. One of the methods suggested to tackle this problem is the use of behavioral models [3], [4]. Behavioral models capture the overall functionality of the circuit in terms of equations or simple circuit elements that are faster to simulate compared to the complete transistor level circuit. Designers often use the basic circuit sizing infrastructure to size the individual blocks and then do a system level simulation using the behavioral models. Designs have to go through this loop a number of times before the final tape-out. This methodology is obviously *not* the best way to design large, complex analog systems.

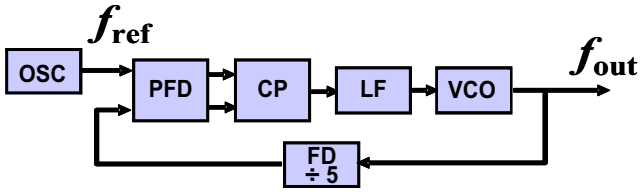

Figure 1 Block level schematic of the PLL

2.2 Proposed methodology

We suggest in this paper a methodology for automatic sizing of circuit elements while keeping system level design specifications under consideration. This is done by (a) using behavioral models for simulating circuits faster, (b) creating pareto trade-off models to capture the capabilities of the circuit across the whole design space and (c) optimizing the system level specs using performance behavioral models for individual blocks. We use a phase–locked-loop as an example to explain our methodology.

Simulating a phase locked loop takes a long time because of the large number of devices in the circuit and the closed loop feedback behavior of the circuit. Hence, calculating the lock-in time requires a large number of CPU cycles. Also, the phase noise specification for the PLL requires a transient noise simulation of the circuit with a small and very well controlled time step and extremely tight tolerances for the simulator to capture the effect of small noise sources. This takes considerable time to simulate--days or weeks. [3] proposes a method of behavioral modeling for PLLs which is efficient while simulating the complete system. The noise models for the individual blocks e.g., VCO, PFD-CP, etc., are captured by simulating the particular circuit block individually. Behavioral models for the individual circuit blocks are created which capture circuit functionality as well as noise behavior. These models are now used for simulating the complete PLL. Since the models are designed in such a way that they insert the noise only at specific instances (zero crossings of the waveforms), it is not very expensive to simulate the whole circuit *including* the noise. Thus, the behavioral models make it possible to evaluate a design point for the PLL in a reasonable amount of time.

Since we have a method of creating the behavioral models which simulates much faster than the transistor level circuit, we can use the simulator-in-the-loop approach to synthesize the PLL. But, the new problem is that the behavioral models represent the circuit characteristics only at *one* specific design point where they were generated (modeled). At a *different* design point, the same behavioral models are *not* valid and we would need to construct a new set of models. Since the modeling procedure is expensive, we cannot afford to create models at each design point. To circumvent this problem, we create a *pareto* surface for the critical circuit block across its design space and use this pareto surface to optimize for the system level specifications.

2.3 Behavioral modeling

Figure 1 shows the block level diagram of the PLL. The circuit is a 0.18um CMOS 500MHz digital frequency synthesizer using a 100MHz reference signal. All the individual blocks--the phase frequency detector, charge pump, frequency divider and VCO--were behaviorally modeled based on [3]. The loop filter (Figure 2) is simply a

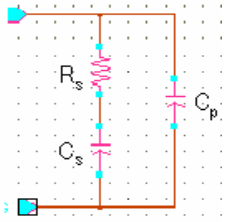

Figure 2 Schematic of the loop filter

2^{nd} order RC filter and hence was not abstracted since modeling it will not give us much in terms of simulation speed-up.

The voltage controlled oscillator (Figure 3) was one of the more challenging blocks to model. Not only did we have to capture the output frequency behavior of the circuit as a function of the control voltage, but also the dynamics observed at the output with abrupt changes at the input. To capture the output frequency behavior as a function of control voltage, we used an equation of the form

$$f_{out} = F(V_{ctrl}) \quad (1)$$

where F is a polynomial equation of degree three. The coefficients for the non-linear equation are found using regression on the simulation data. Basically, the output frequency of the VCO is computed for a series of voltage values at the control voltage. Using this data, we find the coefficients of the polynomial equation via fitting. The dynamic behavior of the VCO is captured by adding a pole at the input of the VCO model. The parameters of the pole (R and C in parallel) are obtained by training the model against the circuit response for a square wave input control voltage. Figure 4 shows the model of the VCO. A similar procedure was adopted for modeling the functionality of all the other blocks as well.

Once we created the behavioral models for the individual blocks as suggested in [3], we simulated the modeled PLL and the transistor level circuit to check the efficacy of the model. We observed a close agreement between the responses of the two with significant simulation speed-ups for the model. The time required for simulating the PLL for 3µs was about 2 hours for the transistor level circuit. It was just 30 seconds for the behavioral models. Also, the behavioral models simulated the noise as well and predicted the phase noise behavior of the PLL which could not be done for the transistor level circuit without waiting for about a week for doing the transient noise simulation.

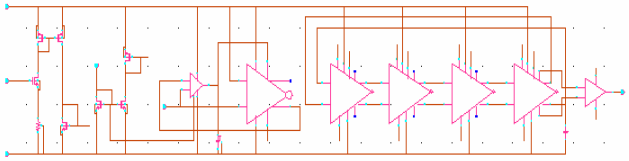

Figure 3 Current starved ring oscillator VCO schematic

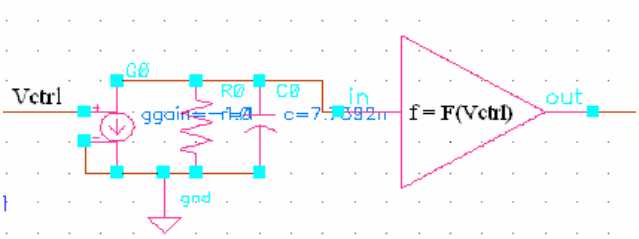

Figure 4 Behavioral model for VCO

2.4 Pareto surface modeling

The VCO is a very important block in a PLL. It is a major contributor to the overall phase noise of the PLL. We want to find the optimum sizing for the transistors in the VCO so that it best meets the overall PLL specifications. As discussed earlier, we cannot directly synthesize the VCO circuit "outside" of the PLL. Since the amount of jitter produced by a VCO often trades-off with the power it consumes, we created a pareto curve that captures the trade-off between jitter and power for the VCO across its complete design space. Following are the steps involved in the creation of the pareto curve [5]:

- The pareto trade-off curve was generated with bias current (power) and jitter as the two variables.
- The other performance specifications for the VCO, for example, the minimum and maximum attainable frequency, linearity, area etc., were treated as constraints which need to be satisfied for any feasible point on the pareto.
- Two feasible design points were found through synthesis, one that minimized power and the other that minimized jitter.
- Using these two points, we used the synthesis tool to look for a design point which minimized both jitter and power such that for some small value ε,

  | Normalized jitter – Normalized power| < ε (2)
- Additional points were found using synthesis that lay *between* these prior points using the same approach as described in the previous step.

Using these points (that lie on the pareto) we fit a non-linear equation which captures the trade-off between jitter and power for VCO across the design space (see resulting curve in Figure 5).

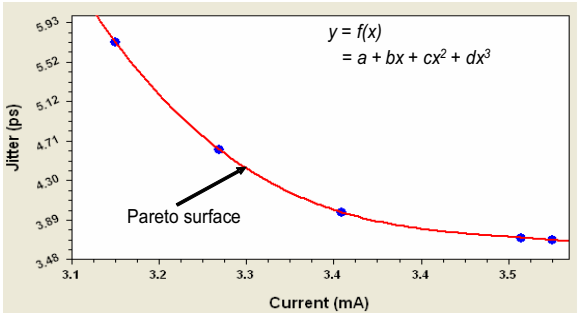

Figure 5 Pareto bias(power)/jitter tradeoff for VCO

We note that in equation (2) the 'ε' term, which has the form of a "slack" specification, gives some flexibility to the synthesis tool to look at a large number of solution points and minimize the performance spec, instead of keeping on searching for an exact numerical equality for tradeoffs.

We can use the pareto curve and the simplified trade-off equations in the behavioral model for the VCO which to represent the oscillator's response across the *whole* design space (Figure 6). Since the phase noise and settling time behavior for the PLL are strongly dependent on the response of the loop filter and the VCO, we can use the circuit elements of the loop filter and performance variables for the VCO as design variables to optimize the overall PLL specifications.

```
module vcoModel (out, currentOut, in);
...
analog begin
  @(initial_step) begin
    // Pareto equation for jitter trade-off with power
    jitter = (1.03-(.87*cur)+(.24*cur^2)-(.02*cur^3))*1p;
  end

  //Output freq. of the VCO as function of control volt.
  freq = (-1.27 + 9.99*cos(1.10*V(in) + 4.29))*1e8;
...
  // Adding phase noise to the output
  dT = 1.414*jitter*$dist_normal(seed,0, 1);
  freq = freq/(1 + dT*freq);
...
  V(out) <+ transition(Vout, 0, tt);
I(currentOut) <+ cur;
```

Figure 6: Verilog-A code for the VCO model

3 SYSTEM LEVEL SIMULATION

All the behavioral models are combined together including the pareto trade-off model for the VCO. Since the behavioral models are much faster to simulate than their circuit level counterparts, it is now easy to look at system level performance trade-offs and optimize the entire PLL. We set up the synthesis run for the PLL using the behaviorally modeled circuit blocks (see Figure 7).

The variables that were used for optimization were R_s, C_s and C_p for the loop filter (Figure 2) and jitter and current for the VCO performance model. The elements of the loop filter were used as design variables because the loop filter plays a crucial role in determine the lock-in time of the PLL. Also, the loop-filter's frequency response shapes the noise from the individual PLL components at the PLL output. All the other circuit blocks were held as fixed. Table 1 shows the specs achieved by the circuit after optimization using commercial analog sizing tools [2,7] from Neolinear.

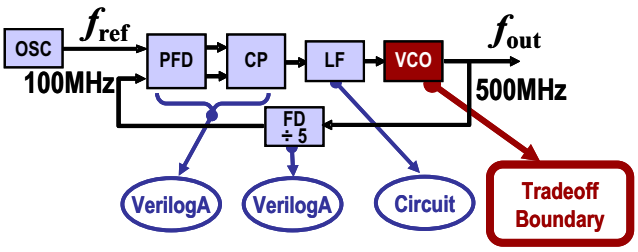

Figure 7 PLL model(s) used for system synthesis

Performance Specification	Achieved Value
PLL Settling Time	0.822 us
VCO bias current	3.37 mA
PLL jitter	0.45% (9.1ps)

Table 1: PLL performance specs after optimization

The PLL was simulated for 4000 cycles at each synthesis point and it took about 4 hours to synthesize the optimal circuit solution on a single CPU. The optimized variable values for the PLL include jitter and current for the VCO. We thus need to search for a design point for the VCO which would have the given jitter and current spec as obtained through PLL synthesis. Since the performance model for the VCO was built using the feasible pareto boundary for these two variables, it will not be very difficult to find a VCO design point that has the given jitter and current specs. This was verified by another synthesis run which obtained the widths and lengths of the transistors used in the VCO that satisfy *these* jitter and current requirements. In order to verify that our final design does indeed meet the specs for the PLL, we simulated the final transistor level circuit and its equivalent macromodel. Figure 8 shows the waveforms at the input control voltage of the voltage controlled oscillator for both the circuits. As can be seen from figure, the responses of the two circuits

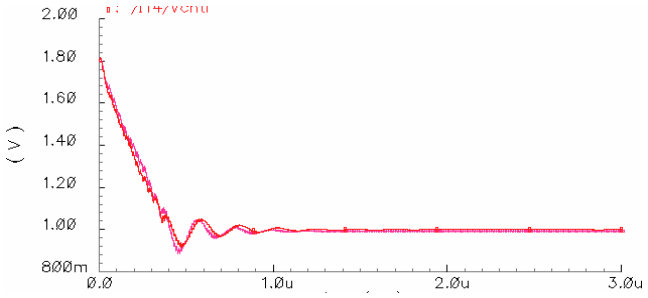

Figure 8 Comparison plot of the control voltage of the VCO for complete PLL simulation showing both transistor level circuit and efficient behavioral model.

match each other quite closely which confirms the accuracy of our models.

4 CONCLUSION

Simulation-based synthesis uses efficient global optimization to visit many circuit candidates, and fully evaluates each candidate solution via detailed simulation. The time to simulate difficult specifications of a PLL is the serious bottleneck: specifications such as jitter can easily require hours or days. The obvious solution is to replace each circuit in the PLL—the frequency divider, VCO, phase-frequency detector, charge pump, loop filter—with an appropriate macromodel. The problem is that we require more than just a model of one circuit instance. We need a model of the *entire* set of performance tradeoffs feasible for the circuit, since our goal is to synthesize each sub-circuit to optimize the performance of the *entire* PLL.

In this work, we showed how to adapt existing synthesis methods to build *pareto-optimal* tradeoff curves that represent the space of achievable specifications for each circuit in a block we wish to optimize. PLL synthesis selects *where* on the curve each circuit needs to be, to optimize the overall PLL behavior, which we evaluate by detailed circuit simulation. Optimization results for one 0.18um CMOS PLL achieved a jitter of 9.13ps (0.45%), a settling time of 0.82μs, and a VCO bias current of and 3.37mA, evaluated by simulation. In future work, we would like to replace *all* the behavioral models with their pareto optimal trade-off curves and optimize the *complete* system. This would give a globally optimal system level design for the circuit under consideration.

ACKNOWLEDGEMENTS

This work was funded in part by the DARPA NeoCAD program.

REFERENCES

[1] Phelps, M.J. Krasnicki, R.A. Rutenbar, L.R. Carley, J.R. Hellums, "A case study of synthesis for industrial-scale analog IP: Redesign of the equalizer/filter frontend for an ADSL CODEC," Proc. DAC'00, June 2000.

[2] A. H. Shah, S. Dugalleix, and F. Lemery. "Technology migration of a high performance CMOS amplifier using an automated front-to-back analog design flow," Proc. DATE'02, Mar. 2002.

[3] K. Kundert, "Predicting the phase noise and jitter of PLL based frequency synthesizers," http://www.designers-guide.com

[4] T., K. Ogawa, K. Kundert, "VCO jitter simulation and its comparison with measurement," Proceedings of the ASP-DAC '99. Asia and South Pacific, Jan. 1999.

[5] I.Das, J.E. Dennis, "Normal boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems," SIAM J. of Optimization, Vol 8,No 3, August 1998.

[6] D. Vasilyev, M. Rewinski, J. White, "A TBR-based trajectory piecewise-linear algorithm for generating accurate low order models for non-linear analog circuits and MEMS," DAC, June 2003.

[7] http://www.neolinear.com