Joint Design-Time and Post-Silicon Optimization for Digitally Tuned Analog Circuits

Wei Yao, Yiyu Shi, Lei He and Sudhakar Pamarti
Electrical Engineering Dept., University of California, Los Angeles, CA, 90095, USA
{weiyao, yshi, lhe, spamarti}@ee.ucla.edu

***Abstract*— Joint design time and post-silicon optimization for analog circuits has been an open problem in literature because of the complex nature of analog circuit modeling and optimization. In this paper we formulate the co-optimization problem for digitally tuned analog circuits to optimize the parametric yield, subject to power and area constraints. A general optimization framework combing the branch-and-bound algorithm and gradient ascent method is proposed. We demonstrate our framework with two examples in high-speed serial link, the transmitter design and the phase-locked-loop (PLL) design. Simulation results show that compared with the design heuristic from analog designers' perspective, joint design-time and post-silicon optimization can improve the yield by up to 47% for transmitter design and up to 56% for PLL design under the same area and power constraints. To the best of the authors' knowledge, this is the first yield-driven analog circuit design technique that optimizes post-silicon tuning together with the design-time optimization.**

I. Introduction

As process technologies scale down to 90nm and below, traditional circuit design methodologies are confronted by the prominent problem of process variation. To deal with process variation for analog circuits, which are highly sensitive to device matching, traditional corner-based design is adopted to guarantee performance in the worst-case scenarios at the cost of substantial circuit overhead. Such corner-based design methodology, however, is becoming insufficient and may eventually be inviable as variation increases with technology scaling.

Statistical design is proposed to analyze the performance distribution from process variation and defines *parametric yield* as the probability the design meets a specified performance or power constraint. Different techniques exist to maximize the parametric yield for analog circuits and generally fall into two complementary categories: *design-time optimization* and *post-silicon tuning*.

Design-time optimization techniques explore the design space at system-level and device-level to maximize the yield for analog circuits. At system-level, different circuit architectures are explored for a trade-off between power, area, and performance. Moreover, some architectures such as closed-loop negative feedback have good immunity from process variation. On the other hand, the impact of process variation can also be reduced by device-level optimization such as transistor sizing [1] and layout optimization. Design-time optimization, however, has difficulty covering all process corners in a cost efficient fashion and may result in high area/power overhead.

Post-silicon tuning in analog design has been widely adopted to combat process variation. Tunable elements such as programmable capacitance array (PCA) [2] and resistance array are proposed to adjust analog circuit performance after chip fabrication [3], [4]. Fig. 1 shows two examples of the tunable elements in analog design: tunable CMOS current source and capacitance array, where β is the resolution (number of control bits). By applying appropriate, potentially different, control signals $D[i]$ $(1 \leq i \leq \beta - 1)$ on individual chips, performance can be adjusted to maximize yield. While this will be discussed in more detail in Section II, we would like to point out that in both examples the tuning values are digitized. Such *digitally tuned analog circuits* have wide applications because of their noise-insensitivity and good technology scalability [5].

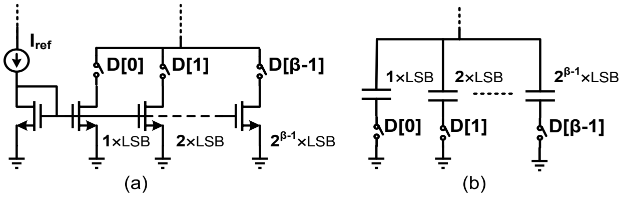


Fig. 1. Examples of digitally tuned analog circuits: (a) CMOS current source and (b) capacitance array.

Post-silicon tuning has been shown to directly impact the design-time optimization for analog circuits [5]. On one hand, post-silicon tunability can significantly relax the analog design by providing a certain capacity to "correct" performance deviation after fabrication. On the other hand, tuning circuitry consumes extra area and power which needs to be considered during design-time optimization in order to meet design specifications. The strong coupling between design-time optimization and post-silicon tuning has already led to joint optimization in the respective domains of both digital circuit design [6] and high-level synthesis [7]. It is natural to expect that by extending joint design-time and post-silicon optimization to analog design, a better parametric yield can be achieved. The complication of modeling and optimizing tunable analog circuits, however, leaves co-optimization an open problem in literature.

In this paper, we study the joint design-time and post-silicon optimization with focus on digitally tuned analog circuits. This type of circuit has two special properties: first, variables such as the transistor sizes are continuous, while variables such as the tuning resolution are discrete in nature. Second, if the resolutions are the only changing variables and all the remaining variables are fixed, we can show that finding the performance upper bound among all permissible resolutions is easy. To make use of these two properties, we propose a general optimization framework combining the branch-and-bound algorithm on the resolutions and gradient-ascent method on the unpruned branches. We use the high-speed serial link as our application and provide two analog design examples to demonstrate the joint optimization framework: transmitter equalization filter design and phase-lock loop (PLL) design. In the transmitter design, we use the transistor sizes, number of taps, resolution, and the least significant bit (LSB) size of the pre-emphasis filter as the optimization variables and propose mathematical models of bit error rate (BER), power, and area with respect to those variables. Our experimental results show that compared with the design heuristic commonly used by analog designers, joint design-time and post-silicon optimization can improve the yield by up to 47% under the same area and power constraints. The same framework is applied to a tunable PLL as another example. We use the charge pump currents as our design variables and formulate the problem to maximize the yield defined by output clock jitter. Result shows the jitter yield can be improved by up to 56% with power and area constraints when compared with

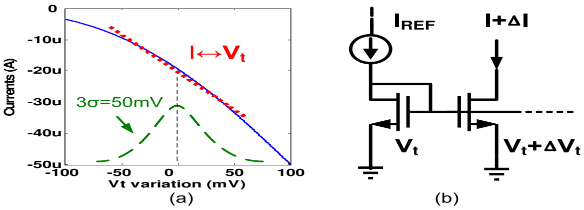


Fig. 2. V_{th} variation model (a) and current mirror with V_{th} mismatch (b).

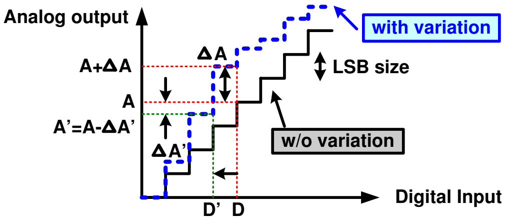


Fig. 3. Post-silicon tuning through DAC

the design heuristic. To the best of the authors' knowledge, this work is the first yield-driven analog circuit design technique that considers post-silicon tuning and design-time optimization at the same time.

The remaining of the paper is organized as follows: Section II briefly reviews the post-silicon tuning technique and Section III provides the formulation for our joint optimization problem. Section IV discusses the proposed optimization framework which combines the branch-and bound technique and gradient ascent method. The designs for the transmitter and PLL circuits in high-speed serial link are discussed in Section IV and V. Experimental results are presented in Section VI and concluding remarks are given in Section VII.

II. Preliminaries on Digitally Tuned Analog Circuits

Analog circuits are very sensitive to process, voltage, and temperature (PVT) variations. Among all sources of variations, the random mismatches caused by doping fluctuations are expected to become dominant within the next few technology generations. In this paper, we focus on the transistor threshold voltage(V_{th}) mismatch and use it as our main source of process variation. Fig. 2 shows an example of threshold voltage(V_{th}) variation and the resulting transistor drain current mismatch. The relation between the V_{th} variation and the resulting drain current I_D can be linearly approximated [8] as

$$I_D = I_{D0} + \eta \Delta V_{th}, \tag{1}$$

where η and I_{D0} can be obtained through SPICE simulation, as shown in Fig. 2(a). Such drain current variation then causes significant power and performance variation in analog design.

To address this issue, various analog design techniques are proposed to reduce the impact of variations. In particular, post-silicon tuning is widely used to calibrate process variation after fabrication using tunable elements. Examples of tunable elements can be found in Fig. 1. In those tuning elements, digital binary control signal is adopted because digital signal is not sensitive to noise and as a result, makes itself immune to variation sources. Those digitally tuned analog circuits conceptually operate as a digital-to-analog conversion (DAC) circuit. By given a control signal i.e. D, an analog output i.e. A, is produced proportionally. There are two major design aspects for digitally tuned analog circuits: least-significant-bit (LSB) size and resolution. The LSB size determines the minimum step in the digital-to-analog conversion. In the CMOS current source shown in Fig. 1(a), for example, it physically represents the drain current for the LSB transistor (I_{LSB}). In the capacitance array shown in Fig. 1(b), it represents the minimum size capacitance (C_{LSB}) in the array. Resolution, on the other hand, is the number of bits used as input control signal. Given the LSB size and resolution, the tuning range can be directly determined. In this paper, we denote its resolution as β and the LSB size as γ.

An example of a digital-to-analog conversion curve is shown in Fig. 3. Assume that digital input D is designed to generate analog output A. With the V_{th} variation, however, the conversion curve becomes nonlinear, and input D generates output with a ΔA deviation with respect to A. To make the analog output closer to the desired value, one can change the input from D to D' and, therefore, a smaller deviation $\Delta A'$ can be obtained. In general, post-silicon tuning is performed by increasing or decreasing the input stepwise to find the minimum deviation.

By applying the tuning technique, the effect of process variation can be significantly reduced. Extra circuits, however, are needed to provide tunability. We assume $D = [100]$ and generate $A = 4 \cdot I_{LSB}$ in Fig. 3. In addition to the required 4 LSB current sources, we need to implement a total of 7 LSB current sources to achieve 3-bit tunability, almost doubling the required area. Moreover, extra sources add capacitance and can potentially increase power consumption. Therefore, an optimal balance between the performance and area/power cost considering system design and post-silicon tuning must be found.

III. Problem Formulation

Without loss of generosity, analog design-time optimization can be described to determine the optimal design parameters to maximize the parametric yield, subject to the power and area constraints [1]. Mathematically,

$$\begin{aligned}
(\mathbf{P}0) \qquad \max \qquad & Prob(F(\boldsymbol{x}) \leq \bar{f}) && (2)\\
s.t. \qquad & Prob(P(\boldsymbol{x}) \geq \bar{p}) \leq \epsilon, && (3)\\
& A(\boldsymbol{x}) \leq \bar{a} && (4)\\
& \boldsymbol{x}_l \preceq \boldsymbol{x} \preceq \boldsymbol{x}_u, \quad \boldsymbol{x} \in R^k && (5)
\end{aligned}$$

where $F(\cdot)$, $P(\cdot)$, and $A(\cdot)$ represent the functions of performance metric, area, and power, respectively. $\bar{f}$, $\bar{p}$ and $\bar{a}$ are the upper bounds of the performance metric; power and area given by the design specifications; $\boldsymbol{x}$ is the vector of length k formed by the design variables with lower bound $\boldsymbol{x}_l$ and upper bound $\boldsymbol{x}_u$ given by the design specifications; k is the total number of design variables; ϵ is a small positive number indicating the tolerance for power variation over the upper bound $\bar{p}$. Note that in the above formulation we have assumed process variation has little impact on the area and the cut-off metric for the performance is an upper bound.

With post-silicon tuning, we first consider the special structure of the digitally tuned elements, as shown in Fig. 1. In this paper, we adopt a simple but direct method based on the unit cell design technique for the tunable element. An example of a unit cell for the CMOS current source is shown in Fig. 4(a). Assume that we have characterized a total number of m unit cells with different transistor width/length and bias voltage under the condition that they all draw the same amount of current I_{unit}. Each unit cell α_i represents a set of transistor W/L and bias voltage V_b, where $0 \leq \alpha_i \leq m$. Any larger transistor, which draws larger current and provides larger swing at the output, can be obtained by connecting the unit cells of the same type in parallel. Such parallel connection ensures linear relationship for the parasitic capacitance and current driving capability, which is

[1] Note that we can also formulate the problem to minimize the power with given performance and area constraints. The joint optimization problem to be proposed can be re-formulated accordingly, and the same optimization framework still applies with little change.

measured by output swing and delay as shown in Fig. 4(b). Moreover, by limiting the maximum number of connected cells, the transistor-level biasing constraints can be guaranteed to ensure all transistors work in the desired operation region. Note that similar unit cell design methodology can be extended to other digitally tuned elements, such as capacitance array.

As a result, the parametric yield can be rewritten as

$$Prob(\hat{F}(\boldsymbol{x},\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma}) \le \bar{f}) \tag{6}$$

where $\hat{F}(\cdot)$ is the performance metric after tuning, $\boldsymbol{\alpha}$ are the indices of the types of unit cell design, and $\boldsymbol{\beta}$ are vectors representing the resolution used for the digitally tuned elements. $\boldsymbol{\gamma}$ are the LSB sizes in terms of the number of unit cells used to implement the LSB of the digitally tuned element. In addition, post-silicon tuning also affects the power consumption, and (3) can be rewritten as

$$Prob(\hat{P}(\boldsymbol{x},\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma}) \ge \bar{p}) \le \epsilon, \tag{7}$$

where $\hat{P}(\cdot)$ is the power consumption after tuning.

Combining the above discussion, the joint design-time and post-silicon optimization can be extended from (**P0**) as

$$\begin{array}{llll}
(\mathbf{P}1) & \max & Prob(\hat{F}(\boldsymbol{x},\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma}) \le \bar{f}) & (8)\\
 & s.t. & Prob(\hat{P}(\boldsymbol{x},\alpha,\boldsymbol{\beta},\boldsymbol{\gamma}) \ge \bar{p}) \le \epsilon & (9)\\
 & & A(\boldsymbol{x},\alpha,\boldsymbol{\beta},\boldsymbol{\gamma}) \le \bar{a} & (10)\\
 & & \boldsymbol{x}_l \preceq \boldsymbol{x} \preceq \boldsymbol{x}_u, \quad \boldsymbol{x} \in R^k & (11)\\
 & & 0 \preceq \boldsymbol{\alpha} \preceq m\mathbf{1}, \quad \boldsymbol{\alpha} \in Z^n & (12)\\
 & & \mathbf{0} \preceq \boldsymbol{\beta}, \quad \boldsymbol{\beta} \in Z^n & (13)\\
 & & \mathbf{0} \preceq \boldsymbol{\gamma}, \quad \boldsymbol{\gamma} \in Z^n, & (14)
\end{array}$$

where m is the total number of unit cell designs and n is the total number of tuning elements in the circuit. Note that there is no explicit bound necessary for $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ as they are implicitly bounded by the power and area constraints (9) and (10).

IV. Optimization Framework

(**P1**) is hard to solve in general because it is a mixed integer non-convex programming problem, the complexity of which grows exponentially with the number of integer variables (the dimension of the vectors $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$). Therefore, we propose to separate the integer variables and the continuous variables. We define a new function $Z(\boldsymbol{t})$ as the optimum value of **(P1)** when $\boldsymbol{x} = \boldsymbol{t}$. If (**P1**) is infeasible at $\boldsymbol{x} = \boldsymbol{t}$, then $Z(\boldsymbol{t}) = -\infty$. Accordingly, **P1** is equivalent to an unconstrained nonlinear optimization problem with a continuous feasible region:

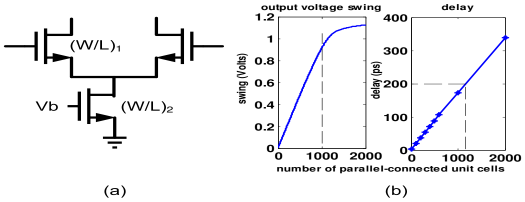
$$\max Z(\boldsymbol{t}), \quad \boldsymbol{t} \in R^k, \tag{15}$$

which can be solved efficiently by the first order gradient method if we can evaluate $Z(\boldsymbol{t})$ and $\frac{\partial Z(\boldsymbol{t})}{\partial \boldsymbol{t}}$ at any point $\boldsymbol{t} = \hat{\boldsymbol{t}}$ to find local maximum. Below we will discuss how to evaluate the function value and first order derivative efficiently.

A. Algorithm Overview

To evaluate $Z(\boldsymbol{t})$ we need to solve problem (**P1**) for given $\boldsymbol{x} = \boldsymbol{t}$, i.e.,

$$\begin{array}{lllll}
(\mathbf{P}2) & Z(\boldsymbol{t}) = & \max & Prob(\hat{F}(\boldsymbol{t},\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma}) \le \bar{f}) & (16)\\
 & & s.t. & Prob(\hat{P}(\boldsymbol{t},\alpha,\boldsymbol{\beta},\boldsymbol{\gamma}) \ge \bar{p}) \le \epsilon & (17)\\
 & & & A(\boldsymbol{t},\alpha,\boldsymbol{\beta},\boldsymbol{\gamma}) \le \bar{a} & (18)\\
 & & & 0 \preceq \boldsymbol{\alpha} \preceq m\mathbf{1}, \quad \boldsymbol{\alpha} \in Z^n & (19)\\
 & & & \mathbf{0} \preceq \boldsymbol{\beta}, \quad \boldsymbol{\beta} \in Z^n & (20)\\
 & & & \mathbf{0} \preceq \boldsymbol{\gamma}, \quad \boldsymbol{\gamma} \in Z^n, & (21)
\end{array}$$



(a)

(b)

Fig. 4. (a) Unit cell design. (b) Swing and delay vs. number of parallel-connected cells.

with variables $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$. (**P2**) is an integer programming problem, which is an NP-hard problem. Though software does exist in literature to solve general integer programming problems, in this paper we propose an optimization framework to efficiently solve it using the special properties of digitally tuned analog circuits.

As delineated in Algorithm 1, the optimization framework combines the branch-and-bound (BnB) algorithm with the gradient ascent method (GDA). Assume that we know how to partition the feasibility space into different regions and how to efficiently obtain an upper bound of the objective function (16) for each region. Then, according to the principles of the BnB algorithm, we can prune regions that have an upper bound worse than the existing solutions, thereby maximizing the performance metric. If a region cannot be pruned, we employ GDA optimization to find a local maximum in it. The final solution $Z(\boldsymbol{t})$ is obtained by comparing the optimal solutions found in each unpruned region.

To evaluate the first order derivative $\frac{\partial Z(\boldsymbol{t})}{\partial \boldsymbol{t}}$, a direct method would be to use the finite different method: Compute $Z(\boldsymbol{t}+\delta\boldsymbol{e_i})$ $(1 \le i \le k)$ for some small positive number δ, where $\boldsymbol{e_i}$ is a unit vector with the i^{th} element equal to 1 and other elements equal to 0. Then the i^{th} element of $\frac{\partial Z(\boldsymbol{t})}{\partial \boldsymbol{t}}$ can be obtained by

$$\frac{\partial Z(\boldsymbol{t})}{\partial t_i} \approx \frac{1}{\delta}(Z(\boldsymbol{t}+\delta\boldsymbol{e_i}) - Z(\boldsymbol{t})). \tag{22}$$

As such, the cost for evaluating $\frac{\partial Z(\boldsymbol{t})}{\partial \boldsymbol{t}}$ would be quite expensive as we would have to solve k integer programming problems. Note that k is the total number of design variables, which can be quite large in practical problems. This urges us to turn to some alternative approach to approximate the computation with affordable cost.

As delineated in Algorithm 1, since we can obtain the upper bound of the objective function in each region efficiently, the upper bound of $Z(\boldsymbol{t})$ is just the maximum of all those upper bounds. Denoting the upper bound of $Z(\boldsymbol{t})$ as $\bar{Z}(\boldsymbol{t})$, the derivative of $Z(\boldsymbol{t})$ can be approximated by applying finite difference method on $\bar{Z}(\boldsymbol{t})$, i.e.,

$$\frac{\partial Z(\boldsymbol{t})}{\partial t_i} \approx \frac{1}{\delta}(\bar{Z}(\boldsymbol{t}+\delta\boldsymbol{e_i}) - \bar{Z}(\boldsymbol{t})). \tag{23}$$

Note that the accuracy of the approximation depends on how the upper bound is calculated. If the upper bound is tight, then the approximation will converge to the exact derivatives.

Next we will discuss how to solve the two critical sub-problems: (**P3**) how to partition the feasible space and derive the upper bound of the objective function for each partitioned region and (**P4**) how to use the GDA method to find a local maximum in each region that cannot be pruned.

B. Partitioning and Bound Estimation

From Algorithm 1 we can see that the BnB+GDA framework offers a trade-off between runtime and quality: a finer partition of the solution space results in fewer local optimums in each region and accordingly, better GDA optimization quality, but at the cost of a increased runtime for the BnB algorithm as the number of total regions increases. In this paper, we partition the solution space according to the unit cell

Algorithm 1 BnB+GDA algorithm framework for computing $Z(\boldsymbol{t})$ and $\frac{\partial Z(\boldsymbol{t})}{\partial \boldsymbol{t}}$.

```
Evaluate (16) to get $\hat{z}$ by initial guess.
(P3): Partition the feasible space $\Omega$ into regions $\omega_i$ ($1 \leq i \leq d$) and derive the upper bound of the objective function $\bar{z}_i$ in each region.
$\bar{Z}(\boldsymbol{t}) = \max_i\{z_i\}$.
for $i = 1$; $i \leq d$; $i$++ do
   if $\bar{z}_i \leq \hat{z}$ then
      Continue;
   else
      (P4): Solve (P2) in $\omega_i$ for optimal value $\tilde{z}_i$ by the GDA method.
      if $\tilde{z}_i \geq \hat{z}$ then
         $\hat{z} = \tilde{z}_i$.
      end if
   end if
end for
$Z(\boldsymbol{t}) = \hat{z}$.
Evaluate $\bar{Z}(\boldsymbol{t} + \delta \boldsymbol{e_i})$ for a small positive number $\delta$.
$\frac{\partial Z(\boldsymbol{t})}{\partial t_i} \approx \frac{1}{\delta}(\bar{Z}(\boldsymbol{t} + \delta \boldsymbol{e_i}) - \bar{Z}(\boldsymbol{t}))$.
```

index and LSB size of each tap. In other words, each region has a unique set of unit cell indices and LSB sizes. Our experiments show that such partitioning provides a good balance between the runtime and the solution quality.

In general, the yield upper bound for a given region is hard to compute. Fortunately, in this particular type of problem, where digitally tuned analog circuits are involved, we are able to obtain the bound through a special relaxation. Suppose we can solve (**P2**) without power and area constraints, then such an optimal value can serve as the upper bound of the constrained problem (**P2**) since we have expanded the feasible space. Note that such an upper bound might not be a tight one since the corresponding solution may violate the area or power constraint.

To solve (**P2**) without constraints, we need to resort to its physical meaning: given the unit cell design and LSB sizes, find the optimal resolution that gives the maximum yield. The optimal resolution can be determined according to the target values for the tuning parameters in an iterative way, as delineated in Algorithm 2. The iterative procedure is required because in most cases the target values are also related to the resolution due to the area-dependent parasitics. In experiments, we find that the algorithm converges quickly within two or three iterations. The optimality of the solution is guaranteed because any increase in the resolution only increases the total area and the parasitics while the minimum distance to the target values remains the same, which will downgrade the performance.

Algorithm 2 Yield upper bound computation for given unit cell design and LSB sizes (**P3**).

```
INPUT: Unit cell indices $\tilde{\boldsymbol{\alpha}}$ and LSB sizes $\tilde{\boldsymbol{\gamma}}$;
OUTPUT: Yield upper bound $\bar{z}$;
INIT: Set initial guess $\beta^{(0)}$; $k = 1$;
while $\max_k |\boldsymbol{\beta}^{(\boldsymbol{k})} - \boldsymbol{\beta}^{(\boldsymbol{k-1})}| > \epsilon||\boldsymbol{\beta}^{(\boldsymbol{k-1})}||$ do
   Calculate the system parasitics according to $\tilde{\boldsymbol{\alpha}}$, $\tilde{\boldsymbol{\gamma}}$ and $\boldsymbol{\beta}^{(\boldsymbol{k})}$;
   Update system response;
   Find the target optimal values for all tuning parameters ;
   Determine $\boldsymbol{\beta}^{(\boldsymbol{k+1})}$ according to $\tilde{\boldsymbol{\alpha}}$, $\tilde{\boldsymbol{\gamma}}$ and the target optimal values;
   $k = k + 1$;
end while
$\bar{z} = Prob(\hat{F}(\boldsymbol{t}, \tilde{\boldsymbol{\alpha}}, \boldsymbol{\beta}^{(\boldsymbol{k})}, \tilde{\boldsymbol{\gamma}}) \leq \bar{f})$;
```

C. Gradient Ascent Method

Given the partitioning method discussed in the previous section, if a particular region cannot be pruned by comparing its upper bound with the current solution, we need to solve (**P2**) for optimal $\boldsymbol{\beta}$ with given unit cell indices $\tilde{\boldsymbol{\alpha}}$ and LSB size set $\tilde{\boldsymbol{\gamma}}$.

In essence, the gradient ascent method sequentially takes steps in a direction proportional to the gradient, until a local maximum of the objective function is reached [9]. At each step we increase/decrease each variable by 1 and check the change of the objective function. Note that by doing so we are actually computing the gradient because all the variables are integers. We then move along the direction that causes the maximum increase. This is iteratively done until the relative change of the objective value is below a certain threshold. The termination of the algorithm indicates that one of the local maxima has been reached or that we have reached the boundary. The initial guess for the GDA can be arbitrarily chosen. In our experiments, we found that it did not influence runtime or quality significantly for both of the examples studied. In addition, we observed that the algorithm always converges to local optimum within two or three iterations.

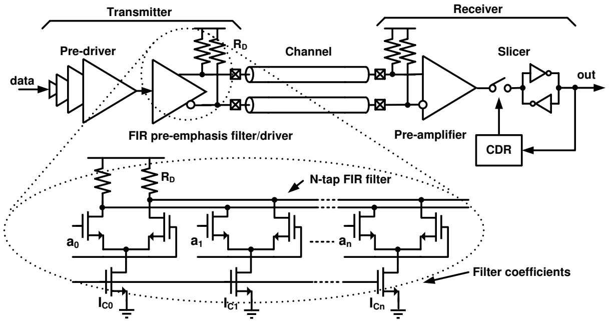


Fig. 5. System diagram of a high-speed serial link.

Next we will use a high-speed serial link as our application and provide two analog design examples, the transmitter design and the phase-lock loops (PLLs) design, to demonstrate our joint optimization framework.

V. Transmitter Design in High-speed Serial Link

The system diagram of a high-speed serial link is shown in Fig. 5. At the transmitter end, the pre-driver drives the FIR pre-emphasis filter at the designated data rate. The pre-emphasis filter is used to counteract the inter-symbol interference (ISI) [10] caused by the bandwidth-limited channel, which behaves as a transmission line and can be characterized by the Telegrapher's equations with RLGC per-unit-length model. At receiver end, the pre-amplifier, along with the slicer decision circuit, is responsible for detecting the data from the received signal. Moreover, the clock is embedded in the transmitted data and the clock data recovery (CDR) sub-system is used to extract the clock from the serial data stream.

In the transmitter design, the pre-emphasis filter plays an important role in both the design quality and the post-silicon tunability [10], thus rendering it a good example for joint design-time and post-silicon optimization as shown later in this section. The pre-emphasis filter can be expressed as

$$b_i = \sum_{j=0}^{n-1} W_j a_{i-j}, \tag{24}$$

where n is the number of filter taps, W_i is the tap coefficient for tap i, and a_i is the transmitted non-return-to-zero (NRZ) symbol. The filter coefficient W_i can be determined adaptively by the least-mean-square (LMS) algorithm [10]. The filter is usually implemented by current-mode logic (CML). The coefficient of each tap is realized by the CMOS current source.

In order to focus on the transmitter optimization, in our first example, we assume that the frequency domain response for the channel and the receiver is given. In addition, we assume that an ideal sampling clock is obtained through the receiver CDR circuits.

Design-time optimization for high-speed serial link transmitter has been well studied in literature. For example, in [11] the tradeoff between bit resolution and power consumption is studied. Recently, a framework for simultaneous circuit-and-system design-space exploration has been proposed for high-speed links [12] in which transmitter optimization is one of the primary targets. In this paper, we use the unit cell design technique for each filter coefficient as shown in Fig. 4.

A. Design-time Optimization

The performance of the overall system is usually quantified in terms of BER, the rate at which errors occur during data transmission. To start with, we formulate the design-time optimization problem to minimize the BER of the system subject to power and area constraints. The design variables include the number of taps n of the filter, the transistors sizing W/L, and the bias voltage V_b in the CMOS current source. Assume that we have characterized a total number of m unit cells and each unit cell α_i represents a set of transistor W/L and bias voltage V_b, as shown in Fig. 4.

Since directly measuring the BER requires a long period of time, error vector magnitude (EVM) is used in this paper to estimate the BER because of their monotonic relationship [13].

$$EVM = \sqrt{\frac{1}{M}\frac{\sum_1^M |r_i - a_i|^2}{|r_{max}|^2}}, \tag{25}$$

where

$$r_i = \sum_{j=-\infty}^{\infty} b_j p_{i-j} + n_i, \tag{26}$$

is the received data with respect to filter output b_i from (24), time domain symbol response p_i, and circuit thermal noise n_i. Moreover, r_{max} is the outermost received data in the constellation and M (usually less than 10^4) is the total number of data used for computation. We can easily map the EVM to the BER from table look-up and accordingly, the objective function (2) takes the form

$$Prob(BER(n, \boldsymbol{\alpha}) \leq \bar{f}). \tag{27}$$

The area $A(n, \boldsymbol{\alpha})$ and power $P(n, \boldsymbol{\alpha})$ of the transmitter are mainly contributed by the pre-emphasis filter and the pre-driver, i.e.

$$A(n, \boldsymbol{\alpha}) = A_{pre-driver}(n, \boldsymbol{\alpha}) + A_{filter}(n, \boldsymbol{\alpha}), \tag{28}$$

$$P(n, \boldsymbol{\alpha}) = P_{pre-driver}(n, \boldsymbol{\alpha}) + P_{filter}(n, \boldsymbol{\alpha}). \tag{29}$$

For tap i $(1 \leq i \leq n)$, we use unit cells of type α_i $(1 \leq \alpha_i \leq m)$ with the parasitic capacitance $C_{unit}^{\alpha_i}$ and the occupied area $A_{unit}^{\alpha_i}$. The required number of cells q_i for that tap is determined by its coefficient W_i and the unit current I_{unit}:

$$q_i = \lceil \frac{W_i}{I_{unit}} \rceil. \tag{30}$$

Accordingly, the total area used in the pre-emphasis filter can be calculated as

$$A_{filter}(n, \boldsymbol{\alpha}) = \sum_{i=1}^{n} q_i A_{unit}^{\alpha_i}. \tag{31}$$

The total parasitic capacitance C_{para} can be calculated as

$$C_{para}(n, \boldsymbol{\alpha}) = \sum_{i=1}^{n} q_i C_{unit}^{\alpha_i}. \tag{32}$$

The power consumed by the filter (P_{filter}) contains both static power and dynamic switching power and can be expressed as

$$P_{filter}(n, \boldsymbol{\alpha}) = \rho \sum_{i=1}^{n} q_i \cdot I_{unit} \cdot V_{dd} + (1-\rho) f \cdot V_{dd}^2 \cdot C_{para}, \tag{33}$$

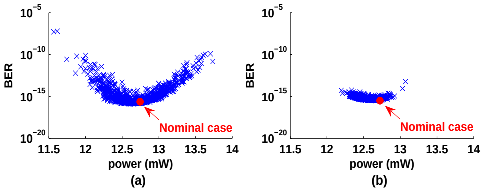


Fig. 6. Power and performance variation for 1000 die samples by Monte Carlo simulation: (a) without tuning and (b) with tuning.

where f is the data rate. ρ is the ratio between static power and total power, which depends on detailed delay and switching probability and can be obtained from simulation.

The pre-driver is designed according to the total gate capacitance at the filter input $C_{gate} = \sum_{i=1}^{n} q_i C_g^{\alpha_i}$, where $C_g^{\alpha_i}$ is the input transistor gate capacitance of unit cell α_i. We assume the pre-driver is designed through logic effort using a simple inverter chain. Note that other configurations like CML pre-drivers with swing control can also be applied. As a result, the occupied area can be determined by

$$A_{pre-driver} = A_{inv} \cdot (1 + f_p + \cdots {f_p}^{N_p}), \tag{34}$$

where $N_p = ln\lfloor \frac{C_{gate}}{C_{inv}} \rfloor$ and $f_p = (\frac{C_{gate}}{C_{inv}})^{\frac{1}{N_p}}$. A_{inv} and C_{inv} are the area and input capacitance for a unit inverter and N_p is the number of pre-driver stages. The pre-driver consumes only dynamic power:

$$P_{pre-driver} = \frac{1}{2} f \cdot vdd^2 \cdot C_{inv} \cdot (1 + f_p + \cdots {f_p}^{N_p}). \tag{35}$$

Combining (27)-(35), the optimization problem can then be mathematically formulated as shown in (**P0**).

B. Post-silicon Tuning and Joint Optimization

In the presence of process variation, assuming transistor threshold voltage V_{th} has a normal distribution with 10% variation [14], the power consumed by the transmitter varies by 30% variation and the BER varies in the magnitude of $10^8\times$ for the same design, as demonstrated in Fig. 6(a). By applying the tuning technique, simulation results show that the span of power and BER variation becomes much smaller as shown in Fig. 6(b). Extra circuits, however, are needed to provide this tunability and an optimal balance between the performance and area/power cost has to be found.

To cast the problem into the format of (**P1**), we need to find $\hat{F}(\cdot)$, $\hat{A}(\cdot)$ and $\hat{P}(\cdot)$. The $\hat{F}(\cdot)$ is straightforward to obtain:

$$\hat{F} = BER(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}), \tag{36}$$

where $\boldsymbol{\alpha}$ is the vector indicating the LSB design for each tap. $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ are vectors in R^n containing resolution and LSB size for each tap, and $\bar{e}$ is the allowed BER upper bound. Note that the number of taps n is no longer a variable: by allowing $\beta_i = 0$, tap i is removed. Accordingly, we only need to specify n_{max}, a maximum number of taps to be considered ($n_{max} = 10$ in this paper).

The power P_{filter} (33) and area A_{filter} (31) of the pre-emphasis filter also need to be modified with the introduction of the DAC:

$$P_{filter} = \rho \sum_{i=1}^{n_{max}} \boldsymbol{D}_i{}^T [2^{\beta_i - 1}, \cdots, 2^0] \cdot \gamma_i I_{unit} \cdot V_{dd} + (1-\rho) f \cdot V_{dd}^2 \cdot C_{para}, \tag{37}$$

$$A_{filter}(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) = \sum_{i=1}^{n} 2^{\beta_i} \gamma_i A_{unit}^{\alpha_i}, \tag{38}$$

$$C_{para} = \sum_{i=1}^{n_{max}} 2^{\beta_i} \gamma_i C_{unit}^{\alpha_i}. \tag{39}$$

Note that vector $\boldsymbol{D_i}$ represents the digital control bits and P_{filter} becomes a distribution instead of a deterministic value because of the I_{unit} variation from V_{th} mismatch. The other calculations are kept

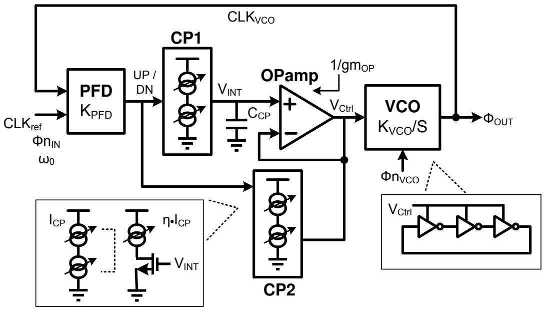


Fig. 7. Tunable and adaptive bandwidth PLL. [17]

the same and the total area and power can be obtained by (28) and (29), accordingly.

VI. PLL Design in High-speed Serial Link

Phase locked-loops (PLLs) are widely used to generate well-timed on-chip clocks in high-speed transceivers [15]. Any timing jitter or phase noise significantly degrades the performance of the system, especially as operating frequency increases.

Timing jitter can be expressed as $\sigma_{\Delta T} = (T/2\pi) \cdot \sigma_{\Delta \phi}$, where ω_0 is the clock frequency, $T = 2\pi/\omega_0$ is the clock period, and $\sigma_{\Delta\phi}$ is the phase jitter of the clock. Phase jitter is defined as the standard deviation of the phase difference between the first cycle and mth cycle of the clock [16].

An example of second-order PLL as shown in Fig. 7 comprises of several components: (1) the phase frequency detector, (2) the charge pump, (3) the loop filter, and (4) the voltage-controlled oscillator. Phase and frequency detector is used to detect phase and frequency difference and provides the UP/DN signal to the charge pump. The charge-pump circuit comprises of two switches driven by the UP and DN signal and injects the charge into or out of the loop filter capacitor (C_{CP}). The combination of charge-pump and C_{CP} is an integrator that generates the average voltage of UP (or DN) signal, V_{Ctrl}, and adjusts the frequency of the subsequent oscillator circuit. In Fig. 7, a power-supply regulated ring oscillator is shown with the voltage-to-frequency gain K_{VCO}. The VCO output frequency is controlled by its supply voltage V_{Ctrl}.

A. Design-time Optimization

The performance of PLL is measured by its output clock jitter. The jitter mainly comes from the reference clock (N_{in}) and VCO (N_{VCO}), which can be expressed as [16]:

$$Jitter = \sigma^2_{\Delta T} = \frac{8}{w_0^2}\int_0^\infty S_\phi(f) sin^2(\pi f \Delta T) df, \tag{40}$$

and

$$S_\phi(f) = \frac{N_{in}}{f^2}\cdot|Hn_{in}(j2\pi f)|^2 + \frac{N_{VCO}}{f^2}\cdot|Hn_{VCO}(j2\pi f)|^2. \tag{41}$$

Note that Hn_{in} and Hn_{VCO} are the noise transfer functions of the reference clock noise (N_{in}) and VCO noise (N_{VCO}) accordingly. Here we assume white noise sources and ignore the noise from the clock buffers.

Considering the PLL shown in Fig. 7, the noise transfer functions Hn_{in} and Hn_{VCO} can be expressed using PLL design parameters:

$$\begin{aligned} Hn_{in}(s) = \frac{\phi_{out}}{\phi n_{in}} &= \frac{K_{loop}RC_{CP}s + K_{loop}}{s^2 + K_{loop}RC_{CP}s + K_{loop}} \\ &= \frac{2\zeta\omega_n s + \omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \end{aligned} \tag{42}$$

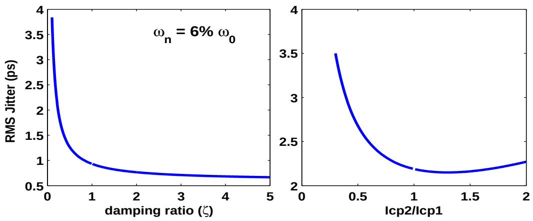


Fig. 8. Output jitter sensitivity to the (a) loop damping factor ζ and (b) charge pump current ratio I_{CP2}/I_{CP1}.

$$\begin{aligned} Hn_{VCO}(s) = \frac{\phi_{out}}{\phi n_{VCO}} &= \frac{s^2}{s^2 + K_{loop}RC_{CP}s + K_{loop}} \\ &= \frac{s^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}, \end{aligned} \tag{43}$$

where $K_{loop} = I_{CP1}/(2\pi C_{CP})K_{PD}K_{VCO}$, $\omega_n = \sqrt{K_{loop}}$, $\zeta = \sqrt{K_{loop}}RC/2$, $R = (I_{CP1}/I_{CP2})(1/gm_{OP})$ [16]. We can see that the noise from the input reference clock and VCO are filtered through low-pass and high-pass filters, respectively.

As a result, the jitter performance is a function of the PLL design parameters ω_n and ζ. Fig. 8(a) shows an example of output root-mean-square (RMS) jitter with respect to the damping ratio (ζ) for a fixed $\omega_n = 0.06\omega_0$. Moreover, in the case of tunable PLL shown in Fig. 7, the natural frequency varies proportionally to $\sqrt{I_{CP1}}$ and the damping factor is proportional to $I_{CP2}/\sqrt{I_{CP1}}$ [17]. By finding an optimum value of the absolute value and relative ratio of I_{CP2} and I_{CP1}, we can minimize the PLL output jitter. We write the objective function of the design-time optimization as (2):

$$Prob(Jitter(\boldsymbol{\alpha}) \le \bar{f}), \tag{44}$$

where $\boldsymbol{\alpha}$ is a vector which represents the number of unit cells used in the charge pumps. In other words, it represents the value of I_{CP1} and I_{CP2}. An example of the relation between output RMS jitter and the current ratio for the charge pumps (I_{CP2}/I_{CP1}) is shown in Fig. 8(b), with a fixed I_{CP1}.

For the design-time optimization, we want to minimize the output clock jitter, subject to power and area constraints. The design parameters are the charge pump currents I_{CP1} and I_{CP2}. The power consumption of the charge pump can be calculated by an approach similar to the one used in our first transmitter design example. Assume we use unit cells of type α_i ($1 \le \alpha_i \le m$) with unit current I_{α_i}, then the required number of cells q_i for the charge pump i can be determined by

$$q_i = \lceil \frac{I_{CPi}}{I_{\alpha_i}} \rceil. \tag{45}$$

As a result, the power consumed by the charge pump is:

$$P_{CP}(\boldsymbol{\alpha}) = \sum_i \frac{1}{2}(2\pi\omega_0)\cdot(1 + \frac{1}{\eta_i})\cdot q_i I_{\alpha_i}\cdot V_{dd}, \tag{46}$$

where η_i represents the current mirror ratio for the biasing circuit of the charge pump i. The area can also be approximated using the similar method and details can be found in our first example. As a result, the optimization problem is mathematically formulated as shown in (**P0**). Note that the power dissipated by PLLs is often a small fraction of total active power. However, it can be quiet significant during sleep modes where the PLL must remain locked.

B. Post-silicon Tuning and Joint Optimization

In the presence of process variation, the output RMS jitter varies for the same design because of the variations on I_{CP1} and I_{CP2}, as demonstrated in Fig. 9(a). To reduce the impact of process variation and improve the parametric yield, post-silicon tuning techniques can

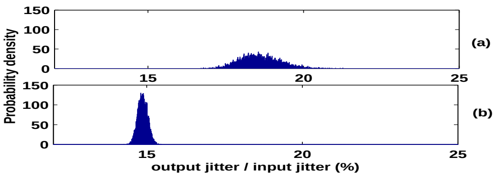


Fig. 9. Probability density for output jitter(%). (a) without tuning (b) with tuning circuit and optimized digital control.

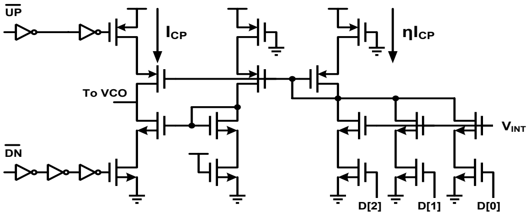


Fig. 10. Charge pump schematic [17].

be applied. Fig. 10 shows a schematic of the charge pump circuit with digitally tuned elements placed in the biasing circuit. By applying a proper digital control signal $\boldsymbol{D}$, the charge pump current ratio can be optimized to reduce the output jitter under the impact of process variation. The resulting histogram can be found in Fig. 9(b).

As discussed in Section III, we can change the objective function to the *Jitter parametric yield* as

$$Prob(Jitter(\boldsymbol{\eta},\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma}) \leq \bar{f}), \tag{47}$$

where $\boldsymbol{\alpha}$ is the vector indicating the LSB design for each tap in the tunable element. $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ contain resolution and LSB size for each charge pump and $\boldsymbol{\eta}$ represents the biasing current ratio; $\bar{f}$ is the allowed jitter upper bound. The power consumed by the charge pump can be re-written as

$$P_{CP} = \sum_i \frac{1}{2}(2\pi\omega_0)\cdot(1+\frac{1}{\eta_i})\cdot \boldsymbol{D}_i{}^T[2^{\beta_i-1},\cdots,2^0]\cdot\gamma_i I_{\alpha_i}\cdot V_{dd}.$$

Note that in this example, the tunable element is inserted in the biasing part with bias ratio η, which is considered as part of the design parameters $\boldsymbol{x}$ in (**P1**). When $\eta \ll 1$, only a small amount of current in the biasing circuit is required to generate I_{CP}. As a result, the power consumed and the area occupied by the digitally tuned element can be ignored. In this case, however, the LSB size in the charge pump current becomes $\frac{1}{\eta}\gamma I_\alpha$, which is increased when η is decreased. The effect of tuning is reduced and may not provide the desired yield. On the other hand, when $\eta \sim 1$, the tunability is maximized but the power and area consumed by the tunable element is also increased. Obviously, a good balance needs to be found through our proposed framework.

VII. Experimental Results

We extract the model parameters by SPICE simulation in IBM 90nm technology and implement the proposed algorithm in MATLAB. All the experiments are run on a Windows server with Pentium IV 3.2GHz CPU and 2G RAM.

A. Transmitter Design

We compare our algorithm with three different methods: no-tunability design, maximum tunability design, and design heuristic from designer's perspective. The design heuristic is guided by the designers' experience [18], [19]: (1) total number of filter taps is iteratively determined by the channel response and the LMS algorithm. (2) assume that each tap of the filter has the same LSB size; (3) the LSB size is determined by considering the maximum and minimum filter coefficient. This design methodology serves as a heuristic for this joint optimization problem and essentially solves the problem in a reduced solution space. The no-tunability design sets the resolution to be 1 ($\beta_i = 1$) for all taps and maximizes the precision of a pre-set pre-emphasis filter. The maximum tunability design uses only one-tap filter ($n_{max} = 1$) to allow maximum adjustability. The no-tunability design and maximum tunability design also serve as the representative of maximum design-time effort and maximum post-silicon effort, respectively.

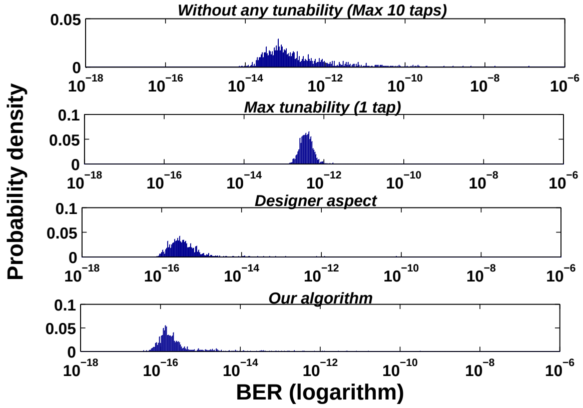


Fig. 11. BER distribution for four different designs.

For fair comparison, the data rate for all the designs is set to be $5GHz$ and the threshold BER for yield $\bar{e} = 1.0 \times 10^{-15}$. In our experiments, we assume that the channel is a 30cm differential microstrip line on FR-4 substrates and that the receiver has ideal timing recovery. We also assume that V_{th} variation follows normal distribution.

We first present the BER distribution with 20% V_{th} variation based on $10K$ Monte Carlo runs in Fig. 11. The area is constrained to $1000um^2$ and the power is constrained to $10mW$. First, for all the four methods, the distributions show strong non-symmetry and non-Gaussianity. This should be attributed to the non-linear relationship between the V_{th} and BER. Second, we can see that the ranges of BER vary for the four methods: the maximum and minimum tunability design gives the smallest and largest variations respectively, with the other two methods in between. This is in accordance with the intuition that more tunability corresponds to less variation. Third, we can see that our design gives the smallest mean BER while the minimum tunability design gives the largest mean BER. Moreover, compared with the design heuristic, our design optimizes the BER distribution with better mean and smaller variance. This verifies that our joint design-time and post-silicon optimization can significantly improve performance when compared with design-time or post-silicon only optimization, or the heuristic method.

Next, we quantitatively study how the yield from our design and design heuristic vary with respect to different area constraints for fixed power ($P = 10mW$) and 20% V_{th} variation. [2] The yield is defined as the percentage of the chips meeting the BER as in (6). The results are presented in Fig. 12 (a). From the figure we can see that for different area specs, our design always gives a larger yield than the design heuristic. Moreover, with the tightening of the area spec, the yield degradation of our method is slower than the design heuristic. When the area is limited to $700um^2$, we have a 47% yield improvement over the design heuristic. Finally, it is interesting to

[2] The BER distributions from our design and design heuristic are better than those from the no/maximum tunability designs in orders of magnitude, thus rendering the yield of the latter two designs close to zero for the same threshold. Accordingly we exclude them for the quantitative comparison.

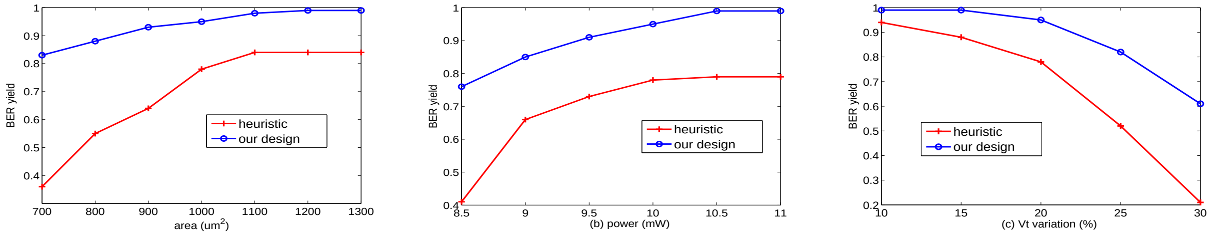


Fig. 12. Yield curves for our designs and design heuristic with respect to area (a), power (b) and V_{th} (c).

note the area saturation effect: When the area constraint is larger than $1200um^2$, the yield does not improve because the design is dominated by the power constraint. We observe that for the $10mW$ power limit, the optimized design area cannot exceed $1200um^2$, regardless of the maximum area allocated. This verifies our discussion that the power and area constraints are strongly coupled.

A similar study is conducted with respect to different power constraints for fixed area ($A = 1000um^2$) and 20% V_{th} variation as shown in Fig. 12 (b). From the figure we can see that for different power specifications, our design also gives better yield and better scalability than the design heuristic. When the power is limited to $8.5mW$, we have a 35% yield improvement over the design heuristic. The power saturation effect is also observed here when the area constraint becomes dominant. In addition, we study how the amount of V_{th} variation affects the yield for the four methods for fixed power ($P = 10mW$) and area ($A = 1000um^2$) constraints. Although V_{th} variation is not explicitly listed as a constraint and only appears in the power and area constraints, it affects the yield significantly. Our design improves the yield by 40% when compared with the design heuristic with 30% variation, as shown in Fig. 12 (c). In terms of runtime, the developed framework is very efficient. For different power and area constraints, the runtime varied between 30 minutes and 1 hour.

B. PLL Design

The same optimization framework is applied to a PLL design example and the result is provided in Fig. 13. We compare our algorithm with the design heuristic that has optimal I_{CP1} and I_{CP2} values through design time optimization and tunable elements in the biasing circuit consumes negligible power [16]. The reference clocks of the PLL for both designs are set to $700MHz$. We assume that the V_{th} variation follows normal distribution. The yield is defined as the percentage of the chips meeting the jitter requirement, as in (47). The experiment was conducted with respect to different power constraints for fixed area and 30% V_{th} variation, as shown in Fig. 13(a). From the figure we can see that for different power specs, our design provides better yield than the design heuristic and obtains up to 29% yield improvement. In Fig. 13(b), when the power is limited to $17mW$, we have a 56% yield improvement over the design heuristic.

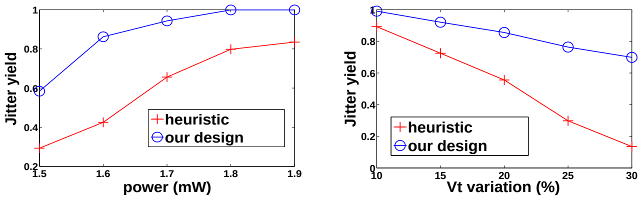


Fig. 13. Yield for our algorithms and design heuristic w.r.t power (a) and V_{th} (b) in the PLL.

VIII. Conclusions

Joint design time and post-silicon optimization for analog circuits has been an open problem in literature, given the complex nature of analog circuit modeling and optimization. In this paper we formulate a co-optimization problem for digitally tuned analog circuits to optimize the parametric yield, subject to power and area constraints. A general optimization framework combing the branch-and-bound algorithm and gradient ascent method is proposed. We demonstrate our framework with two examples in high-speed serial link, the transmitter design and the phase-locked-loop (PLL) design. Experimental results show that compared with the design heuristic from analog designers' perspective, joint design-time and post-silicon optimization can improve the yield by up to 47% for transmitter design and up to 56% for PLL design under the same area and power constraints.

References

[1] M. Pelgrom, A. Duinmaijer, and A. Welbers, “Matching properties of mos transistors,” *J. Solid-State Circuits*, 1989.

[2] H. Darabi, S. Khorram, H.-M. Chien, M.-A. Pan, S. Wu, S. Moloudi, J. Leete, J. Rael, M. Syed, R. Lee, B. Ibrahim, M. Rofougaran, and A. Rofougaran, “A 2.4-ghz cmos transceiver for bluetooth,” *Solid-State Circuits, IEEE Journal of*, vol. 36, pp. 2016–2024, Dec 2001.

[3] H. Huang and E. K. F. Lee, “Design of low voltage cmos continuous-time filter with on-chip automatic tuning,” *J. Solid-State Circuits*, 2001.

[4] G. Miller, M. Timko, H.-S.Lee, E. Nestler, M. Mueck, and P. Ferguson, “Design and modeling of a 16-bit 1.5msps successive approximation adc with non-binary capacitor array,” *Proc. Int. Great Lakes Symp. on VLSI*, 2003.

[5] B. Murmann and B. Boser, “Digitally assisted analog integrated circuits,” *Queue*, vol. 2, no. 1, pp. 64–71, 2004.

[6] M. Mani, A. K. Singh, and M. Orshansky, “Joint design-time and postsilicon minimization of parametric yield loss using adjustable robust optimization,” in *Proc. Int. Conf. on Computer Aided Design*, 2006.

[7] F. Wang, X. Wu, and Y. Xie, “Variability-driven module selection with joint design time optimization and post-silicon tuning,” in *Proc. Asia South Pacific Design Automation Conf.*, 2008.

[8] B. Razavi, *Principles of Data Conversion System Design*. John Wiley and Sons, 1995.

[9] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambrdige University Press, 2004.

[10] Y. Tao, W. Bereza, R. Patel, S. Shumarayev, and T. Kwasniewski, “A signal integrity-based link performance simulation platform,” *Proc. Custom Integrated Circuits Conference*, 2005.

[11] R. Gupta and A. O. Hero, “Power versus performance tradeoffs for reduced resolution lms adaptive filter,” *Trans. On Signal Processing*, 2000.

[12] R. Sredojevic and V. Stojanovic, “Optimization-based framework for simultaneous circuit-and-system design-space exploration: A high-speed link example,” in *Proc. Int. Conf. on Computer Aided Design*, 2008.

[13] S. Sen, V. Natarajan, R. Senguttuvan, and A. Chatterjee, “Pro-vizor: Process tunable virtually zero margin low power adaptive rf for wireless systems,” in *Proc. Design Automation Conf*, June 2008.

[14] Y. Ye, F. Liu, S. Nassif, and Y. Cao, “Statistical modeling and simulation of threshold variation under dopant fluctuations and line-edge roughness,” *Proc. Design Automation Conf*, June 2008.

[15] B. Razavi, *Monolithic Phase-Locked Loops and Clock Recovery Circuits: Theory and Design*. John Wiley & Sons, 1996.

[16] M. Mansuri and C.-K. Ken, “Jitter optimization based on phase-locked loop design parameters,” *Solid-State Circuits, IEEE Journal of*, vol. 37, pp. 1375–1382, Nov 2002.

[17] S. Sidiropoulos, D. Liu, J. Kim, G. Wei, and M. Horowitz, “Adaptive bandwidth dlls and plls using regulated supply cmos buffers,” pp. 124–127, 2000.

[18] J. VIta, A. Marques, P. Azevedo, and J. Franca, *Design Considerations for a Retargetable 12b 200MHz CMOS Current-Steering DAC*. Springer US, 2003.

[19] A. C. Y. Lin and M. J. Loinaz, “A serial data transmitter for multiple 10gb/s communication standards in 0.13um cmos,” in *Int. Solid State Circuits Conf.*, 2008.