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

Abstract— **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 address this problem through an example of high-speed transmitter design. We formulate the co-optimization problem so as to maximize the BER yield subject to the area and power constraints for a given channel and receiver design. An efficient optimization framework combining the branch-and-bound algorithm and gradient ascent method is proposed. Experimental results show that compared with a manual design approach 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. 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 [1]. To deal with process variation for analog circuits, which performance is highly sensitive to the device matching, traditional corner-based design is adopted to guarantee the performance in the worst-case scenarios at the cost of substantial circuit overhead. Such corner-based design methodology, however, is becoming insufficient and may not be viable eventually as the variation increases along with the technology scaling.

Statistical design, as a result, is proposed to analyze the performance distribution from process variation and defines *parametric yield* as the probability of the design meeting a specified performance or power constraint [2], [3]. Different techniques exist to maximize the parametric yield for analog circuits, and these techniques 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 tradeoff between power, area, and performance. Moreover, some architectures such as closed-loop negative feedback has good immunity from the process variation. On the other hand, the impact of process variation can also be reduced by device-level optimization, such as transistor sizing [4], 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 in order to combat process variation. Tunable elements such as the programmable capacitance array (PCA) [5] and resistance array are proposed to adjust the analog circuit performance after chip fabrication [6], [7]. 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 different control signals $D[i]$ $(1 \le i \le \beta - 1)$ on individual chips, the performance can be adjusted to maximize the 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 digitalized. Such *digital tuned analog circuits* have wide applications because of their noise-insensitivity and good technology scalability [8].

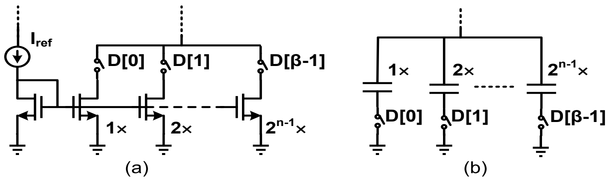

Fig. 1. Examples of digitally tuned analog circuits: (a) CMOS current source and (b) capacitance array.

It has been shown that post-silicon tuning has direct impacts on the design-time optimization for analog circuits. [8]. On the one hand, post-silicon tunability can significantly offload the constraints on analog designs by providing the capacity of "correcting" the performance deviation due to process or evniromental variations to certain extent. On the other hand, the tuning circuitry consumes extra area and power which needs to be considered in the design-time optimization in order to meet the design specs. The strong coupling between design-time optimization and post-silicon tuning has already led to joint optimization both in the domain of digital circuit design [3] and high-level synthesis [9]. It is natural to expect that by extending joint design-time and post-silicon optimization into analog design, a better parametric yield can be achieved. The complication of modeling and optimizing tunable analog circuits, however, leaves the co-optimization an open problem in literature.

In this paper, we study the joint design-time and post-silicon optimization with focus on the digitally tuned analog circuits. This type of circuit has two special properties: First, the variables such as the transistor sizes are continuous, while the variables such as the tuning resolution are discrete in nature. Second, if the resolutions are the only changing variables and all the reamaining variables are fixed, we can show that it is easy to find the performance upper bound among all permissible resolutions. 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 high-speed serial link as our application and provide two analog design examples to demonstrate the joint optimization framework: transmitter filter design and clock and data recovery (CDR) circuit design. In the trasmitter design, we use the transistor sizes, number of taps, resolution, and 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 a manual design approach 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 tunale phase-lock loop (PLL) in the CDR design as another example. We use the charge pump currents as our design variables and formulate the problem as to maximize the yield defined by output clock jitter. Result shows significant improvement on the jitter yield with the power and area constraints. To the best of the authors' knowledge, this is the first yield-driven analog circuit design technique that optimizes post-

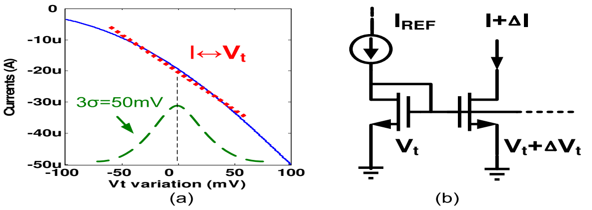

Fig. 2. V_{th} variation model (a) and current mirror with V_{th} mismatch (b).

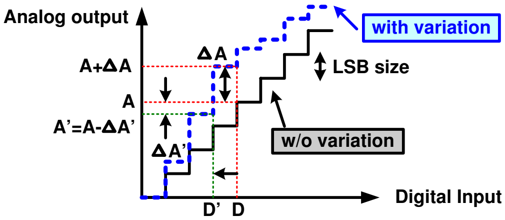

Fig. 3. Post-silicon tuning through DAC

silicon tuning together with the design-time optimization.

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 CDR 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 [1]. In this paper, we consider 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 V_{th} variation and the resulting drain current I_D can be linearly approximated [10] 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 in turn results in significant power and performance variation in analog design.

To address this issue, various analog desgin 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 the noise and, as a result, make itself immunitive against variation sources. Those digitally tuned analog circuits conceptually operate as a digital-to-analog conversion (DAC) circuit. By given a control signal, say D, an analog output, say A, is produced proportionally. There are two major design aspects for thoes digitally tuned analog circuit: 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 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 the general case, post-silicon tuning is performed by increasing or decreasing the input stepwise to find the minimum deviation.

Though by applying the tuning technique, the effect of process variation can be reduced significantly, Extra circuits, however, are needed to provide the tunability. We assume $D = [100]$ and generates $A = 4 \cdot I_{LSB}$ in Fig. 3. In addition to the required 4 LSB current sources, we need to implement total 7 LSB current sources to achieve 3-bit tunability, almost doubling the required area. Therefore, it is important to find an optimal balance between the performance and area/power cost considering system design and post-silicon tuning.

III. Problem Formulation

Without loss of generosity, analog design-time optimization can be described as to find out the optimal design parameters to maximize certain performance metric, subject to the power and area constraints [1]. Mathematically, it can be stated as

$$\begin{aligned} (\mathbf{P0}) \qquad \max \qquad & F(\boldsymbol{x}) && (2)\\ s.t. \qquad & P(\boldsymbol{x}) \leq \bar{p} && (3)\\ & A(\boldsymbol{x}) \leq \bar{a} && (4)\\ & \boldsymbol{x}_l \preceq \boldsymbol{x} \preceq \boldsymbol{x}_u, \quad \boldsymbol{x} \in R^k && (5) \end{aligned}$$

where $F(\cdot)$, $A(\cdot)$, and $P(\cdot)$ represent the functions of performance metric, area, and power, respectively. $\bar{p}$ and $\bar{a}$ are the power and area upper bound given by the design specs. $\boldsymbol{x}$ is the vector formed by the design variables with lower bound $\boldsymbol{x}_l$ and upper bound $\boldsymbol{x}_u$ given by the design specs, and k is the total number of design variables.

In the presence of process variation, we can change the objective function (10) to be the *parametric yield* as

$$Prob(F(\boldsymbol{x}) \leq \bar{f}), \tag{6}$$

where $\bar{f}$ is the allowed performance bound. In addition, the power also has variation due to process variation, and accordingly the power constraint (11) should be re-casted as

$$Prob(P(\boldsymbol{x}) \geq \bar{p}) \leq \epsilon, \tag{7}$$

where ϵ is a small positive number indicating the tolerance for power variation over the upper bound $\bar{p}$.

With the post-silcon 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 unit cell for the CMOS current cource 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

[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.

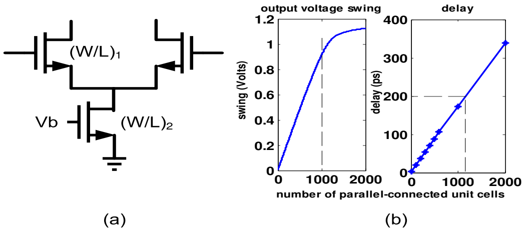

Fig. 4. (a) Unit cell design. (b) Swing and delay vs. number of parallel-connected cells.

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 $0 \le \alpha \le m$. 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 measured by the output swing and the 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 working on 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

$$Prob(\hat{F}(\boldsymbol{x}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) \le \bar{f}) \tag{8}$$

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 (7) can be rewritten as

$$Prob(\hat{P}(\boldsymbol{x}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) \ge \bar{p}) \le \epsilon, \tag{9}$$

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

$$\begin{aligned}
(\mathbf{P}1) \qquad \max \quad & Prob(\hat{F}(\boldsymbol{x}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) \le \bar{f}) && (10)\\
s.t. \quad & Prob(\hat{P}(\boldsymbol{x}, \alpha, \boldsymbol{\beta}, \boldsymbol{\gamma}) \ge \bar{p}) \le \epsilon && (11)\\
& A(\boldsymbol{x}, \alpha, \boldsymbol{\beta}, \boldsymbol{\gamma}) \le \bar{a} && (12)\\
& \boldsymbol{x}_l \preceq \boldsymbol{x} \preceq \boldsymbol{x}_u, \quad \boldsymbol{x} \in R^k && (13)\\
& 0 \preceq \boldsymbol{\alpha} \preceq m\mathbf{1}, \quad \boldsymbol{\alpha} \in Z^n && (14)\\
& \mathbf{0} \preceq \boldsymbol{\beta}, \quad \boldsymbol{\beta} \in Z^n && (15)\\
& \mathbf{0} \preceq \boldsymbol{\gamma}, \quad \boldsymbol{\gamma} \in Z^n, && (16)
\end{aligned}$$

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 explict bound necessary for $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ as they are implicitly bounded by the power and area constraint (11) and (12).

IV. Optimization Framework

(**P1**) is hard to solve in general because it is a mixed integer non-convex programming, 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 as follows. 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 continuous feasible region as follows

$$\max Z(\boldsymbol{t}), \quad \boldsymbol{t} \in R^k, \tag{17}$$

which can be solved efficiently by 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.,

$$\begin{aligned}
(\mathbf{P}2) \qquad Z(\boldsymbol{t}) = \max \quad & Prob(\hat{F}(\boldsymbol{t}, vec\alpha, \boldsymbol{\beta}, \boldsymbol{\gamma}) \le \bar{f}) && (18)\\
s.t. \quad & Prob(\hat{P}(\boldsymbol{t}, \alpha, \boldsymbol{\beta}, \boldsymbol{\gamma}) \ge \bar{p}) \le \epsilon && (19)\\
& 0 \preceq \boldsymbol{\alpha} \preceq m\mathbf{1}, \quad \boldsymbol{\alpha} \in Z^n && (20)\\
& \mathbf{0} \preceq \boldsymbol{\beta}, \quad \boldsymbol{\beta} \in Z^n && (21)\\
& \mathbf{0} \preceq \boldsymbol{\gamma}, \quad \boldsymbol{\gamma} \in Z^n, && (22)
\end{aligned}$$

with variables $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$. (**P2**) is an integer programming, 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 making use of the special properties of the 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 (18) 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 maxing 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.

Moreover, 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. Denote the upper bound of $Z(\boldsymbol{t})$ as $\bar{Z}(\boldsymbol{t})$, then the derivative of $Z(\boldsymbol{t})$ can be approximated by applying finite difference method on $\bar{Z}(\boldsymbol{t})$, i.e.,

$$\frac{\partial Z(\boldsymbol{t})}{\partial \boldsymbol{t}} = \lim_{k \to 0} \frac{1}{k} (\bar{Z}(\boldsymbol{t} + k\Delta \boldsymbol{t}) - \bar{Z}(\boldsymbol{t})), \tag{23}$$

where $\Delta \boldsymbol{t}$ is a unit vector point at the direction of derivative. 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 in detail how to solve the two critical subproblems: (**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 tradeoff between the runtime and quality: a finer partition of the solution space results in fewer local optimums in each region and accordingly better GDA optimization quality, at the cost of a increased runtime for BnB algorithm as the number of total regions are increased. In this paper, we partition the solution space according to the unit cell index and LSB size of each tap. In other words, each region has a unique set of unit cell indice and LSB sizes.

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

```
Select an initial solution and evaluate (18) to get $\hat{y}$.
(P3): Partition the feasible space $\Omega$ into regions $\omega_i$ ($1 \leq i \leq d$) and derive the upper bound of the objective function $\bar{y}_i$ in each region.
$\bar{Z}(\boldsymbol{t}) = \max_i\{y_i\}$.
for $i = 1$; $i \leq d$; $i$++ do
  if $\bar{y}_i \leq \hat{y}$ then
    Continue;
  else
    (P4): Solve (P2) in $\omega_i$ for optimal value $\tilde{y}_i$ by the GDA method.
    if $\tilde{y}_i \geq \hat{y}$ then
      $\hat{y} = \tilde{y}_i$.
    end if
  end if
end for
$Z(\boldsymbol{t}) = \hat{y}$.
Evaluate $\bar{Z}(\boldsymbol{t} + k\Delta\boldsymbol{t})$ for a small positive number k.
$\frac{\partial Z(\boldsymbol{t})}{\partial \boldsymbol{t}} = \frac{1}{k}(\bar{Z}(\boldsymbol{t} + k\Delta\boldsymbol{t}) - \bar{Z}(\boldsymbol{t}))$.
```

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 since the objective function in (**P1**) is non-convex and non-differentiable. Fortunately, in this particular type of problems where digitally tuned analog circuits are involved, we are able to obtain the bound through a special relaxation. Suppose we can solve (**P1**) without power and area constraints, then such an optimal value can serve as the upper bound of the constrained problem (**P1**) since we have expanded the feasible space. Note that such an upper bound might not be a tight one, as the corresponding solution may violate the area or power constraint.

To solve (**P1**) without constraints, we need to resort to its physical meaning: Given the unit cell design and LSB sizes, find out 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 remain 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 indice $\tilde{\boldsymbol{\alpha}}$ and LSB sizes $\tilde{\boldsymbol{\gamma}}$;
OUTPUT: Yield upper bound $\bar{y}$;
INIT: Set initial resolutions by 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
$\bar{y} = Prob(\hat{F}(\tilde{\boldsymbol{\alpha}}, \boldsymbol{\beta}^{(\boldsymbol{k})}, \tilde{\boldsymbol{\gamma}}) \leq \bar{f})$;
```

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 (**P1**) for optimal $\boldsymbol{\beta}$ with given unit cell indice $\tilde{\boldsymbol{\alpha}}$ and LSB size set $\tilde{\boldsymbol{\gamma}}$.

The essence of the gradient ascent method is to sequentially take steps in a direction proportional to the gradient, until a local maximum of the objective function is reached [11]. As delineated in Algorithm 3, at each step we in turn 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. Then we move along the direction that causes the maximum increase. This is iteratively done until the relative change of the objective value is below certain threshold. The termination of the algorithm indicates that one of the local maxima has been reached or we have reached the boundary.

Algorithm 3 Gradient Ascent Method (**P4**).

```
INPUT: Unit cell indice $\tilde{\boldsymbol{\alpha}}$ and LSB sizes $\tilde{\boldsymbol{\gamma}}$.
OUTPUT: Optimized objective value $\hat{y}$.
Initialize: Select an initial guess $\boldsymbol{\beta}^{(\boldsymbol{0})}$; $y^{(0)} = Prob(\hat{F}(\tilde{\boldsymbol{\alpha}}, \boldsymbol{\beta}^{(\boldsymbol{0})}, \tilde{\boldsymbol{\gamma}}) < \bar{f})$; $k = 1$;
while $|y^{(k)} - y^{(k-1)}| \geq \epsilon|y^{(k-1)}|$ do
  for $i = 1$;$i \leq n$; $i$++ do
    $\beta_i^{(k-1)}$=$\beta_i^{(k-1)} + 1$;
    if (??) and (??) are satisfied for $\boldsymbol{\beta}^{(\boldsymbol{k-1})}$ and $\tilde{\boldsymbol{\gamma}}$ then
      $\Delta_i^+ = Prob(\hat{F}(\tilde{\boldsymbol{\alpha}}, \boldsymbol{\beta}^{(\boldsymbol{k-1})}, \tilde{\boldsymbol{\gamma}}) < \bar{f}) - y^{(k-1)}$;
    else
      $\Delta_i^+$ = -Inf;
    end if
    $\beta_i^{(k-1)} = \beta_i^{(k-1)} - 2$;
    if (??) and (??) are satisfied for $\boldsymbol{\beta}^{(\boldsymbol{k-1})}$ and $\tilde{\boldsymbol{\gamma}}$ then
      $\Delta_i^- = Prob(\hat{F}(\tilde{\boldsymbol{\alpha}}, \boldsymbol{\beta}^{(\boldsymbol{k-1})}, \tilde{\boldsymbol{\gamma}}) < \bar{f}) - y^{(k-1)}$;
    else
      $\Delta_i^-$ = -Inf;
    end if
    $\beta_i^{(k-1)} = \beta_i^{(k-1)} + 1$;
  end for
  $\boldsymbol{\beta}^{(\boldsymbol{k})}$ = Increase/decrease the element in $\boldsymbol{\beta}^{(\boldsymbol{k-1})}$ corresponding to the maximum in $\Delta^+$ and $\Delta^-$;
  $k = k + 1$;
end while
$\hat{y} = y^{(k)}$;
```

The initial guess can be arbitrarily chosen as in our experiments we find that it does not influence the runtime or quality significantly for both of the examples studied. In addition, we observe that the algorithm always converges with less than 0.001% error ($\epsilon = 1e-5$) within two or three iterations.

V. Transmitter Design in High-speed Serial Link

In this paper, we use high-speed serial link as our application and provide two analog design examples to demonstrate the joint optimization framework: transmitter design and clock and data recovery (CDR) circuit design. 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) [12] caused by bandwidth-limited channel, which behaves as transmission line and can be characterized by the Telegrapher's equations with RLGC per-unit-length model [13]. At receiver end, the pre-amplifier, along with slicer decision circuit are 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 [12], 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

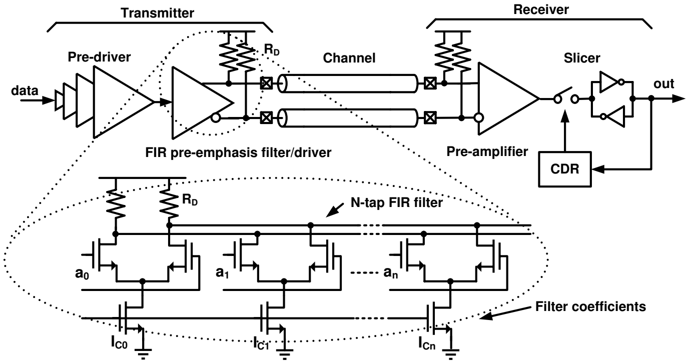

Fig. 5. System diagram of a high-speed serial link.

filter coefficient W_i can be determined adaptively by the least-mean-square (LMS) algorithm [12]. The filter is usually implemented by current-mode logic (CML), and 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. Another example about the CDR circuit optimization is dicsussed in the next section.

A. Design-time Optimization

Design-time optimization for high-speed serial link transmitter has been well studied in literature. For example, in [14] 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 [15] 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.

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 as to minimize the BER of the system subject to the power and area constraints. The design variables include the number of taps n of the filter, the transistors sizing W/L and 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 [16].

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

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.

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

$$A = A_{pre-driver} + A_{filter}, \tag{27}$$

$$P = P_{pre-driver} + P_{filter}. \tag{28}$$

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

$$q_i = \lceil \frac{W_i}{I_{unit}} \rceil. \tag{29}$$

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

$$A_{filter}(n, \boldsymbol{\alpha}) = \sum_{i=1}^{n} q_i A_{unit}^{\alpha_i}, \tag{30}$$

where $\boldsymbol{\alpha}$ is a vector with its element α_i representing the unit cell design used for filter tap i [2]. The total parasitic capacitance C_{para} can be calculated as

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

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

$$P_{filter}(n, \boldsymbol{\alpha}) = \rho \sum_{i=1}^{n} q_i \cdot I_{unit} \cdot V_{dd} + (1-\rho) f \cdot V_{dd}^2 \cdot C_{para}, \tag{32}$$

where f is the data rate. ρ is the ratio between static power and total power, which depends on the 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. As a result, the occupied area can be determined by

$$A_{pre-driver} = A_{inv} \cdot (1 + f_p + \cdots {f_p}^{N_p}). \tag{33}$$

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} is 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:

$$P_{pre-driver} = \frac{1}{2} f \cdot vdd^2 \cdot C_{inv} \cdot (1 + f_p + \cdots {f_p}^{N_p}). \tag{34}$$

Combining (27)-(34), the optimization problem can then be mathematically formulated as the formate 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 [17], 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.

As discussed in Section II, we can change the objective function to be the *BER parametric yield* as

$$Prob(BER(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) \le \bar{e}), \tag{35}$$

[2] From now on, we will use **bold** font to represent a vector.

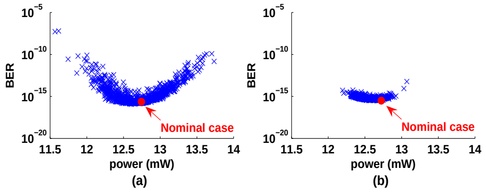

Fig. 6. Power and performance variation for 1000 die samples by Monte Carlo simulation: (a) without tuning and (b) with tuning.

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). Accordingly, such parametric yield evaluates how the system performance deviates from its expectation under process variation after tuning. It can be obtained by Monte Carlo simulation with sampling on V_{th}. Meanwhile, the power P_{filter} (32) and area A_{filter} (30) of the pre-emphasis filter also need to be modified with the introduction of the DAC:

$$P_{filter} = \rho \sum_{i=1}^{n_{max}} \boldsymbol{D}_i{}^T [2^{\beta_i - 1}, \cdots, 2^0] \cdot \gamma_i I_{unit} \cdot V_{dd} + (1-\rho) f \cdot V_{dd}^2 \cdot C_{para}, \tag{36}$$

$$A_{filter}(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) = \sum_{i=1}^{n} 2^{\beta_i} \gamma_i A_{unit}^{\alpha_i}, \tag{37}$$

$$C_{para} = \sum_{i=1}^{n_{max}} 2^{\beta_i} \gamma_i C_{unit}^{\alpha_i}. \tag{38}$$

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 the same and the total area and power can be obtained by (27) and (28), accordingly.

VI. PLL Design in Clock and Data Recovery Circuit

Clock and data recovery circuit (CDR) is a critical function in high-speed transceivers [18]. The data received in these systems are both asynchronous and noisy, requiring that a clock be extracted to allow synchronous operations. Furthermore, the data must be retimed such that the jitter accumulated during transmission is removed.

We now consider a simple CDR circuit [18], as shown in Fig. 7. The purpose of clock recovery circuit is to sense the data and produce a periodic clock, which drives the D flip-flop (DFF) and retimes the data (i.e., it samples the noisy data), yielding an output with less jitter. The generated clock must have a frequency equal to the data rate and it must bear a certain phase relationship with respect to data, allowing optimum sampling of the bits by the clock. As illustrated in Fig. 7, a phase-lock loop (PLL) is employed to generate the clock (CLK_{VCO}) waveform with multiple phases. The clock must exhibit a small jitter since it is the principal contributor to the retimed data jitter. Timing jitter can be expressed as $\sigma_{\Delta T} = (T/2\pi) \cdot \sigma_{\Delta\phi}$, where $T = 2\pi/\omega_0$ 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 [19].

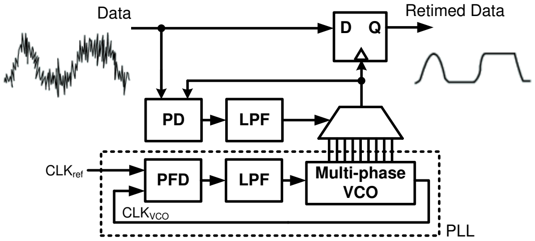

Fig. 7. Block diagram for a clock and data recovery (CDR) circuit.

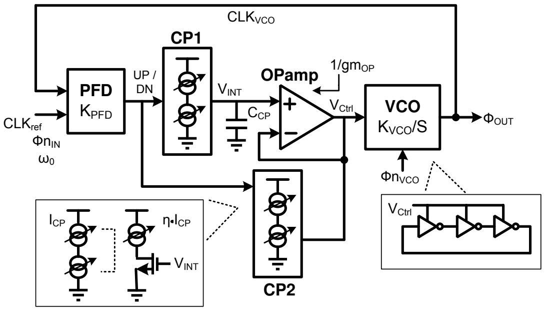

Fig. 8. Tunable and adaptive bandwidth PLL. [20]

An example of second-order PLL is shown in Fig. 8 [20], which is widely used in the high-speed system to generate a low jitter clock. A PLL comprises of several components: (1) phase frequency detector, (2) charge pump, (3) loop filter, and (4) voltage-controlled oscillator. Phase and frequency detector is used to detect the phase and frequency difference and provides the UP/DOWN signal to the charge pump. The charge-pump circuit comprises of two switches driven by the UP and DOWN 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 DOWN) signal, V_{Ctrl}, and adjusts the frequency of the subsequent oscillator circuit. In Fig. 8, 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 [19]:

$$\sigma_{\Delta T}^2 = \frac{8}{w_0^2} \int_0^\infty S_\phi(f) sin^2(\pi f \Delta T) df, \tag{39}$$

and

$$S_\phi(f) = \frac{N_{in}}{f^2} \cdot |Hn_{in}(j2\pi f)|^2 + \frac{N_{VCO}}{f^2} \cdot |Hn_{VCO}(j2\pi f)|^2. \tag{40}$$

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.

Consider the PLL shown in Fig. 8, the noise transfer functions Hn_{in} and Hn_{VCO} can be expressed using PLL design parameters:

$$\begin{aligned} Hn_{in}(s) &= \frac{\phi_{out}}{\phi n_{in}} \\ &= \frac{K_{loop} R C_{CP} s + K_{loop}}{s^2 + K_{loop} R C_{CP} s + K_{loop}} \\ &= \frac{2\zeta\omega_n s + \omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}, \end{aligned} \tag{41}$$

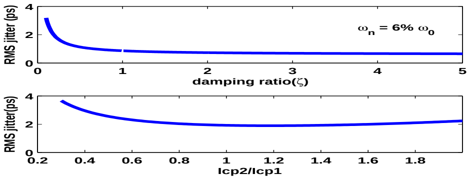

Fig. 9. Output jitter sensitivity to the (a) loop damping factor and (b) charge pump current ratio.

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

where $K_{loop} = I_{CP1}/(2\pi C_{CP})K_{PD}K_{VCO}$, $\omega_n = \sqrt{K_{loop}}$, $\zeta = \sqrt{K_{loop}}RC/2$, $R = (I_{CP1}/I_{CP2})(1/gm_{OP})$. Note 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. Fig. 9(a) shows an example of output root-mean-square (RMS) jitter with respect to damping ratio (ζ) for a fixed $\omega_n = 0.6\omega_0$. Moreover, in the case of tunable PLL shown in Fig. 8, the natural frequency varies proportional to $\sqrt{I_{CP1}}$ and the damping factor is proportional to $I_{CP2}/\sqrt{I_{CP1}}$ [20]. 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. An example of the relation between output RMS jitter and the current ratio for the charge bumps (I_{CP2}/I_{CP1}) is shown in Fig. 9(b), with a fixed I_{CP1}.

For a fixed VCO and PFD design, the design-time optimization problem can be formulated as minimizing the output clock jitter subject to power and area constraints. The design parameters are the charge pump currents I_{CP1} ,I_{CP2} and the current mirror ratio η for the bias current. The power consumption of the charge bump can be calculated by the similar approach used in our first transmitter design example. Assume we use unit cells of type α_i ($1 \leq \alpha_i \leq m$) with unit current I_{α_i}, then the required number of cells q_i for the charge pump i can be determined by

$$q_i = \lceil \frac{I_{CPi}}{I_{\alpha_i}} \rceil. \tag{43}$$

As a result, he power consumed by the charge pump can be expressed as:

$$P_{CP}(\boldsymbol{\eta}, \boldsymbol{\alpha}) = \sum_i \frac{1}{2}(2\pi\omega_0) \cdot (1 + \frac{2}{\eta_i}) \cdot q_i I_{\alpha_i} \cdot V_{dd}. \tag{44}$$

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 can then be mathematically formulated as the formate shown in (**P0**), where $\boldsymbol{\eta}$ can be considered as part of the design paramters $\boldsymbol{x}$.

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. 10(a).

To reduce the impact of process variation and improve the parametric yield, post-silicon tuning technique can be applied. Fig. 11 shows a schematic of charge bump circuit with digitally tuned element are place 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. 10(b)

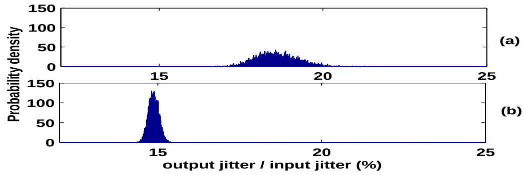

Fig. 10. Probability density for output jitter(%). (a) without tuning (b) with tuning circuit and optimized digital control.

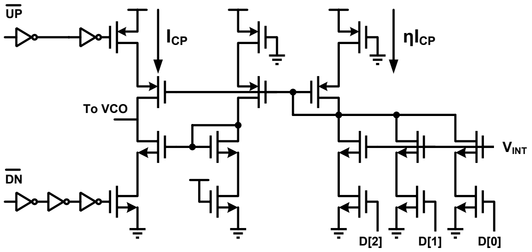

Fig. 11. Charge pump schematic [20].

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

$$Prob(Jitter(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\eta}) \leq \bar{e}), \tag{45}$$

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 bump, and $\bar{e}$ is the allowed jitter upper bound. The power consumed by the charge pump can be re-wrote as

$$P_{CP} = \sum_i \frac{1}{2}(2\pi\omega_0) \cdot (1 + \frac{2}{\eta_i}) \cdot \boldsymbol{D}_i{}^T [2^{\beta_i - 1}, \cdots, 2^0] \cdot \gamma_i I_{\alpha_i} \cdot V_{dd}, \tag{46}$$

Note that in this example, the tunable element is inserted in the biasing part with bias ratio η. 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 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 in this case 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 need to be found through our proposed framework.

VII. Experimental Results

We extract the model parameters by SPICE simulation on a transmitter design 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.

We compare our algorithm with three different methods: manual design, no-tunability design, and maximum tunability design. The manual design is commonly used heuristic guided by designers' experience [21]–[23]: assume that each tap has the same LSB size, which is determined by the minimum value of coefficients. The number of bits for each tap is then calculated according to this LSB size. The manual design serves as a heuristic for joint design-time and post-silicon optimization. The no-tunability design set the resolution to be 1 ($\beta_i = 1$) for all taps such that a maximum number of taps can be achieved, and then solve (**P1**) for *design-time optimization only*. The maximum tunability design uses only one tap ($n_{max} = 1$) to

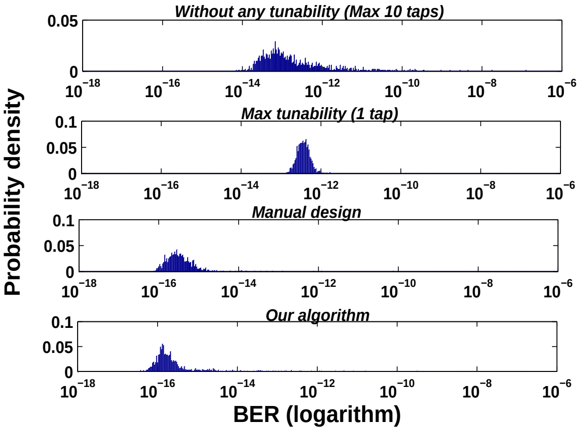

Fig. 12. BER distribution for four different designs.

allow maximum resolutions for all taps and then solve (**P1**) for *post-silicon optimization only*. The no-tunability design and maximum tunability design also serve as the representative of maximum design-time effort and maximum post-silicon effort, respectively.

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 30cm differential microstrip line on FR-4 substrates and the receiver has ideal timing recovery. It is understood, however, that our proposed method is generally applicable to any channels and receivers. 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. 12. 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, better than that of the manual design, while the minimum tunability design gives the largest mean BER. This verifies that joint design-time and post-silicon optimization can significantly improves the performance compared with design-time or post-silicon optimization only.

Next, we quantitatively study how the yield from our design and manual design vary with respect to different area constraints for fixed power ($P = 10mW$) and 20% V_{th} variation [3]. The yield is defined as the percentage of the chips meeting the BER as in (35). The results are presented in Fig. 13 (a). From the figure we can see that for different area specs, our design always gives a larger yield than the manual design. Moreover, with the tightening of the area spec, the yield degradation of our method is slower than the manual design. When the area is limited to $700um^2$, we have a 47% yield improvement over the manual design. Finally, it is interesting to note the area saturation effect: When the area spec 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 actual 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. The results are presented in Fig. 13 (b). From the figure we can see that for different power specs our design also gives better yield and better scalability than the manual design. When the power is limited to $8.5mW$, we have a 35% yield improvement over the manual design. 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. As shown in Fig. 13 (c), 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% compared with the manual design when the variation amount is 30%.

In terms of runtime, the developed framework is very efficient. We observe that for different power and area constraints, the runtime varied between 30 minutes and 1 hour.

VIII. Conclusions and Future Work

In this paper, we use a high speed link transmitter design as an example to study the joint design time and post-silicon tuning optimization. We mathematically formulate the problem as to maximize the BER yield subject to the area and power constraints for given channel and receiver design. An efficient optimization framework combining the branch-and-bound algorithm and gradient ascent method is proposed. Experimental results show that compared with the manual design, joint design time and post-silicon optimization can improve the yield by up to 47% under the same area and power constraints.

In the future, we will further study how joint design-time and post-silicon optimization can be extended to the entire high-speed link, including the channel and receiver.

References

[1] S. I. Associate, *Int. Technology Roadmap for Semiconductors*. 2005.

[2] S. Borkar, T. Karnik, S. Narendra, J. Tschanz, A. Keshavarzi, and V. De, "Parameter variations and impact on circuits and microarchitecture," in *Proc. Design Automation Conf*, 2003.

[3] 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.

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

[5] 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.

[6] 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.

[7] 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.

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

[9] 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.

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

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

[12] 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.

[3] The BER distributions from our design and manual design 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.

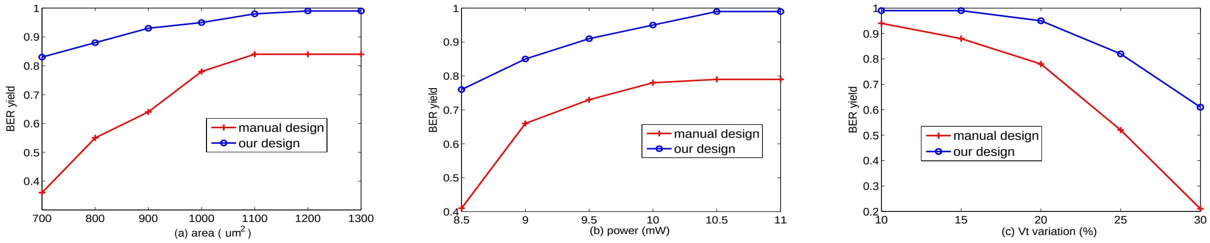

Fig. 13. Yield curves for our designs and manual designs with respect to area (a), power (b) and V_{th} (c).

[13] R. Achar and M. Nakhla, "Simulation of high-speed interconnects," *Proc. of the IEEE*, May 2001.

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

[15] 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.

[16] 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.

[17] 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.

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

[19] 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.

[20] 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.

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

[22] 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.

[23] H. Hernández, W. Noije, E. Roa, and J. Navarro, "A small area 8 bits 50 mhz cmos dac for bluetooth transmitter," *Analog Integr. Circuits Signal Process.*, 2008.