Optimization-based Framework for Simultaneous Circuit-and-System Design-Space Exploration: A High-Speed Link Example

Ranko Sredojević
rasha@mit.edu

Vladimir Stojanović
vlada@mit.edu

Department of Electrical Engineering and Computer Science
Massachusetts Institute of Technology
Cambridge, MA 02139, USA

Abstract—Connecting system-level performance models with circuit information has been a long-standing problem in analog/mixed-signal front-ends, like radios and high-speed links. High-speed links are particularly hard to analyze because of the complex interplay of device/circuit parasitics and channel filtering operation. In this paper we introduce optimization-based framework for link design-space exploration, connecting the link transmission quality and top-level filter settings with circuit power, sizing and biasing. We derive a special analytical discrete time representation that avoids the size explosion of the symbolic problem description improving the parsing and solver time by orders of magnitude and making this joint optimization possible in real-time. This robust and accurate problem formulation is derived in signomial form and is compatible with existing optimization approaches to circuit sizing. We demonstrate this optimization framework on a link design-space exploration example, investigating trade-offs between the transmit pre-emphasis and linear receiver equalizer and their impact on overall link power vs. data rate.

I. INTRODUCTION

In this paper, we aim to bridge the gap between analog/mixed-signal circuit and system design by providing an optimization-based design framework for a fast, simultaneous design space exploration of both system block parameters and underlying circuits, firmly connected to the process technology.

Increasingly tight power and performance constraints mandate a change of the traditional iterative design cycle, in which resource allocation is done at the system level (often using crude circuit/block models and feasibility space intuition from experienced designers) followed by iterations with block/circuit designers to achieve the assigned specifications. The convergence is not guaranteed, often requiring significant and costly re-design time. Additionally, design margins and potential for further improvements are not known in achieved feasible designs. Frequent process migrations further complicate this picture by increasing the design time and cost.

This situation is particularly acute in high-volume analog/mixed-signal front-ends like radios and high-speed link (HSL) I/Os, which have very tight interactions between circuit blocks that are hard to separate at the system level. This is especially the case in HSLs, which are rapidly growing into mini-communication systems due to the bandwidth limitations of chip packages and board traces [1]. Finding best methods to compensate intersymbol interference and minimize timing noise, while running circuits at lowest power and maximum possible data rate is a difficult balancing act that requires extremely tight connection between circuit and system design.

While on one hand accurate HSL system performance models exist [1]–[3], they lack the circuit-level information (power, parasitics, feasible instances and regions of operation) necessary for design-space exploration. On the other hand, simple, intuitive methods, coupled with circuit simulations can sometimes bring further insight into link

design trends [4], but due to simplified performance modeling cannot guarantee feasibility at the system level. To guarantee the feasibility at both levels a joint optimization is required, but has not been reported to date.

In recent years, equation-based circuit optimization has matured beyond initial op-amp examples [5], [6] and grown to the complexity of A/D converters, phase-lock (PLL) and clock-data recovery (CDR) loops [7]–[9]. However, it has not yet reached to larger systems, like a radio or an HSL, mainly due to the lack of tractable system-to-circuit formulations. Here we build on this prior circuit optimization work by providing the system-to-circuit formulation that overcomes the tractability issues and connects the circuit-level parameters with block and system-level specifications, providing a direct link from transistor sizes, biasing and parasitics to system-level HSL metrics like data rate, power and bit-error rate. This formulation enables a simultaneous optimization of multiple-circuits/blocks in an HSL system.

We want this framework to provide answers to questions that link designers often ask: Which equalization method (or combination) should be used (transmit pre-emphasis, linear analog receiver equalizer, decision-feedback equalizer)? Which transmitter signaling method is more power efficient (voltage or current mode)? Should link power be allocated more to the equalizer or to the timing sub-system (PLL and CDR)? What is the power/data-rate trade-off for a given set of channels?

In the following sections we first relate the link performance to the parameters of the transmitter, channel and receiver filters at the system level. We then present the main contribution of this work: a series of transformations of the system model that are necessary to enable tractability of the joint optimization formulation and orders of magnitude improvements in parsing and solver time. We then explain optimization-compliant transistor-level circuit models and use the joint optimization to explore some of the long-standing questions in HSL design, and quickly provide trade-offs such as overall system power vs. data rate and sampling phase.

II. EQUATION-BASED OPTIMIZATION

Extensive work in equation-based circuit optimization resulted in readily available high quality solver engines. In particular, geometric-programming [10] (GP) based signomial solver algorithms, such as [11], provide the main infrastructure for our work. These signomial solver engines (mostly utilizing branch-and-bound with local convexification) provide necessary support for high-quality equation-based circuit and system modeling that relieves some of the GP limitations [12]. Namely, transistor models and biasing constraints can be expressed in signomial form much more accurately than in

314

posynomial [13] form. This is also the case at the system level where modeling different transfer functions would be impossible without the availability of positive and negative terms (e.g. equalizer tap coefficients). It is important to note that circuit and system designers are not removed from this optimization flow, but their time is rather used more efficiently in modeling since designers' expertize and intuition are unmatched model order reduction resources in multi-dimensional circuit design space.

For the HSL example, the joint optimization can be quasi-formally written as

Listing 1 Quasi-formal HSL optimization problem formulation

min	Power
st.	estimate(BER) $\leq BER_{desired}$
	estimate(Area) $\leq Area_{desired}$
	circuit biasing constraints

where power and area estimates come from transistor-level circuit models, while the bit error rate (BER) is estimated at the system level. This joint formulation captures the interactions in which the system-level parameters depend on circuits, and also the "circuit biasing constraints" depend on both "internal" circuit constraints and "outside/environment" operating conditions set by the system parameters.

III. SYSTEM LEVEL MODEL

In Figure 1 we show a block diagram of a typical HSL with the signal processing chain (the transmit-equalizer, the channel, the receive-equalizer filters and the slicer decision circuit), and the timing sub-system.

The BER constraint of an HSL can be expressed as

$$BER = Q(\frac{d_{min}[k] - RX_{Sens}}{\sigma_{Tot}}) < BER_{desired} \tag{1}$$
$$d_{min}[k] > RX_{Sens} + \sigma_{Tot}Q^{-1}(BER_{desired})$$

where $d_{min}[k]$ is the minimum distance between the signal constellation points sampled at phase k and the nearest threshold, and σ_{Tot} is the aggregate rms value of noise, jitter and residual ISI approximated with Gaussian distribution, $BER_{desired}$ is the link target BER, and RX_{Sens} is the input sensitivity of the decision circuit. To improve the accuracy of the BER estimate for $BER_{desired} < 10^{-15}$ (often required in HSLs), the contribution of the residual ISI can be balanced by allocating the N most significant symbol-spaced taps of the symbol response (SR) to the d_{min} and smaller components to σ_{Tot} [14]. In that scenario, $d_{min}[k]$ can be expressed as the sample under worst case destructive interference [15] at the slicer input for a given sampling phase k, and a subset of N most significant symbol-spaced SR taps aligned with phase k

$$d_{min}[k] = h_k - \sum_{j=0,\ j\neq k,\ mod(j-k,ovsRate)=0}^{n-1} |h_j| \tag{2}$$
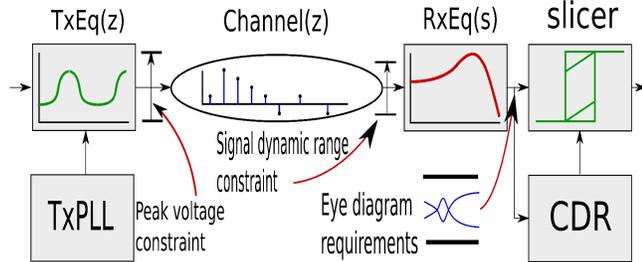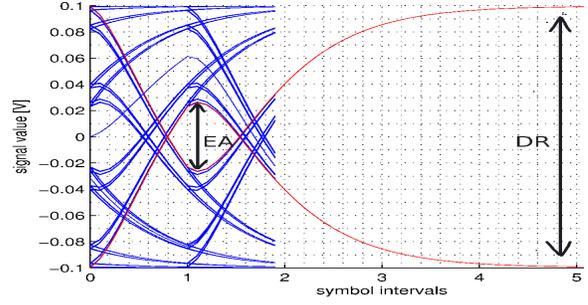$$n = N(ovsRate)$$

Fig. 1.   System level model of a HSL

Fig. 2.   Comparison of a (blue) simulated eye (2^{18} bits) and a (red) calculated worst case eye. Denoted are eye aperture (EA) and dynamic range (DR) of the eye.

where h_k is oversampled equalized SR and $ovsRate$ is the oversampling rate relative to the link symbol-rate.

All remaining SR taps (such as long tails due to reflections at impedance discontinuities) should be treated as additional uncorrelated noise source with rms value of σ_{ISI}

$$\sigma_{Tot}^2 = \sigma_{noise}^2 + \sigma_{jitterV}^2 + \sigma_{ISI}^2 \tag{3}$$

where σ_{noise} is the rms value of the thermal and $\sigma_{jitterV}$ is the rms value of the jitter-induced voltage noise at the receiver [2], [3].

Since the residual ISI dominates the other noise sources in HSLs by an order of magnitude [1], [2], in this work we focus on the optimization of the link signal-processing chain (transmit equalizer and driver, channel, receiver peaking amplifier), noting that the framework can be readily extended to the timing sub-system by using existing jitter-induced voltage noise models [2], [3] and PLL optimization formulations [8].

In Figure 2 we illustrate the values of $d_{min}[k]$ (i.e. the samples that define the eye aperture) along with the dynamic range of the signal DR, calculated from the SR tap values (4). Dynamic range formulation at the system level ensures proper biasing of underlying transmitter and receiver circuits.

$$d_k = \sum_{j=0,\ mod(j-k,ovsRate)=0}^{n-1} |h_j| \tag{4}$$
$$DR = \max_k d_k$$

Since HSLs are usually designed to operate over a range of channels, the link can be simultaneously optimized for a number of channels by adding the constraints (1) and (4) for every channel of interest. The solver performance scales well since increasing the number of constraints in an optimization problem does not significantly affect the performance of interior-point solvers [10].

A. System blocks

In order to formulate (1) and (4) in terms of link block parameters, we first need to obtain the parameterized transfer function of each signal-chain block in Figure 1.

1) The channel: Figure 3 illustrates the frequency response and binary pulse-amplitude modulated SR at $6.25Gb/s$ of a typical 32 inch HSL channel. We represent the channel as an FIR filter

$$C(z) = \sum c_k z^{-k} \tag{5}$$

where c_k are impulse response samples of the channel.

2) Transmit-driver/equalizer (TxEq): Transmit-driver provides impedance matching and power gain to drive the channel while the FIR filter provides adequate equalization to ameliorate severe ISI in HSL channels. Table I describes the $TxEq$ system level parameters.

315

TABLE I
TRANSMIT EQUALIZER (TxEQ) SYSTEM LEVEL VARIABLES

variable name	variable symbol	description
FIR filter coefficients	$[1\ aFR_1\ aFR_2\ \ldots]$	*equalizer coefficients*
output swing	*swingIN*	*maximum transmit swing*

Transmit FIR operates per-symbol and our formulation works with oversampled symbol response, so we upsample the transmit equalizer response and apply a zero-order hold as in (6).

$$txEQ(z;\ swingIN, \mathbf{aFR})) =$$

$$\frac{swingIN}{\sum |aFR_k|}\ \sum_{j=0}^{s-1} aFR_{div(j,\ ovsRate)}\ z^{-j} \tag{6}$$

where $div(a, b)$ is integer division. We normalize the $TxEq$ transfer function in (6) with $||aFR||_1 = \sum |aFR_k|$ to satisfy the output swing limit, since it is convenient to work with $aFR_0 = 1$[1].

3) Receiver equalizer (RxEq): To mitigate ISI, peaking amplifiers [17] are often used in HSL receivers along with decision feedback (DFE) equalizers. The system level parameters of an example peaking $RxEq$ are shown in Table II.

TABLE II
EXAMPLE RECEIVE EQUALIZER (RxEQ) SYSTEM LEVEL VARIABLES

| variable name	variable symbol	description
gain	*scale*	*equivalent Z-domain gain of the system*
zero	*zRX*	*coefficient next to the z^{-1} in the numerator of the transfer function*
drain pole	*dpRX*	*inverse of the pole formed in the drain (in digital domain)*
source pole	*spRX*	*inverse of the pole formed in the source (in digital domain)*

A discrete transfer function approximation is derived[2] as

$$rxEQ(z;\ scale,\ zRX,\ spRX,\ dpRX) =$$

$$scale \frac{1 - zRX z^{-1}}{(1 - spRX z^{-1})(1 - dpRX z^{-1})} \tag{7}$$

.

4) Slicer: The slicer system level model is presented in Table III. The *sensitivity* includes residual offset from process variations and

---

[1] Since GP-derived solver cannot handle negative optimization variables we use a standard linear programming (LP) trick and express each variable of unknown sign as the difference of two positive optimization variables [16]. Absolute value of an expression can be obtained by upper and lower bounding with a variable [16]. These transformations are used for aFR equalization coefficients and h_k SR samples, at the system level.

[2] The discretization will be described in further sections with circuit models.
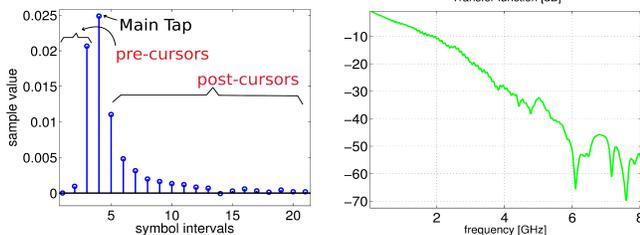
(a) Typical symbol-spaced binary NRZ SR of a channel at $6.25Gb/s$ (b) Typical transfer function magnitude

Fig. 3. A 32 inch off-chip interconnect example

---

overdrive needed for making a proper decision at a given data rate[3].

TABLE III
SLICER SYSTEM LEVEL VARIABLES

variable name	variable symbol	description
input noise	σ_{Tot}	*input referred noise of this block*
sensitivity	RX_{Sens}	*input signal needed for resolving*

IV. SYSTEM LEVEL FORMULATION

With models developed in previous section, and imposed linearity constraints at the circuit level, the HSL system-level model can be represented as a parameterized filter

$$h(z; \mathbf{p}) = txEQ(z; \mathbf{p})C(z)rxEQ(z; \mathbf{p}) = \frac{p(z; \mathbf{p})}{q(z; \mathbf{p})}$$

$$h(z; \mathbf{p}) = \sum_{k=0}^{n-1} h_k(\mathbf{p})z^{-k}, \quad p(z; \mathbf{p}) = \sum_{j=0}^{s-1} p_j(\mathbf{p})z^{-j} \tag{8}$$

$$q(z, \mathbf{p}) = \sum_{j=0}^{r-1} q_j(\mathbf{p})z^{-j}$$

where $\mathbf{p} = (swingIN, aFR, scale, zRX, spRX, dpRX)$ is the system level parameter vector. Equation (8) is a parameterized $\mathcal{Z}$-domain representation of the equalized SR used in (2) and (4). Next we need to transform the parameterized transfer function (8) into parameterized SR samples in (2) and obtain the BER constraint (1) from (2). This parameterized SR expression has to perform well in parsing and solving and it has to be easy to generate automatically due to large number of equations.

Starting from standard time-domain representation of parameterized SR, we first illustrate the dimensionality issues and impracticality of standard time-domain and direct Inverse Discrete Fourier Transform (IDFT) formulations, and present a set of transformations that lead to a very efficient optimization problem formulation that enables orders of magnitude faster parsing and solving, and real-time joint system-and-circuit optimization.

5) Time domain representation: In order to produce signomial equations describing the equalized SR we can first approximate the IIR filters in (8) with FIR filters. For example, we can use Maclaurin series

$$\frac{p(z; \mathbf{p})}{q(z; \mathbf{p})} \approx p(z; \mathbf{p})\tilde{q}(z; \mathbf{p}) \tag{9}$$

where $\tilde{q}(z; \mathbf{p}) = \sum \tilde{q}_k(\mathbf{p})z^{-k}$ is an FIR approximation of the IIR filter $1/q(z; \mathbf{p})$. This approach results in a parameterized expression of each sample of the SR

$$h_k(\mathbf{p}) = \sum_{j=0}^{n-1} p_j(\mathbf{p})\tilde{q}_{k-j}(\mathbf{p}) \tag{10}$$

While this formulation is optimizer-compliant and performs well on smaller problems, it is very limited in practical HSL channels due to its size. An example HSL formulation quickly reveals this curse of dimensionality:

- channel impulse response samples: 20 (bit-spaced)
- IIR filters: 2 filters, 1st order each
- IIR expansion terms: 15 (gives accuracy of 3% for typical pole locations)
- oversampling: 10 (relatively low)

---

[3] Overdrive is inversely proportional to the slicer gain which is an exponential function of the symbol time due to positive feedback in the decision circuit.

since the parameterized SR expression has $(20 \times 10) \times (15 \times 10)^2 = 4,500,000$ terms. The SR is $(20 \times 10) + (15 \times 10) + (15 \times 10) = 500$ samples long, where each sample consists of many terms. This example is a bit optimistic, since our link model should have at least 2 IIR filters and 2 FIR filters. With this complexity of expressions the time to generate the system level description as well as the parsing time are on the order of days, even for this simple example, rendering this approach impractical.

6) Direct IDFT: The next logical approach to formulating the system level is to directly manipulate the expressions in $\mathcal{Z}$-domain. Calculating the overall SR from (8) is done by performing IDFT in n points $z_0, \dots, z_{n-1}$ on the unity circle in $\mathcal{Z}$-plane

$$h_k(\mathbf{p}) = \frac{1}{n} \sum_{j=0}^{n-1} \frac{p(z_j; \mathbf{p})}{q(z_j; \mathbf{p})} z_k^j, \quad k = 0, \dots, n-1 \tag{11}$$

which has fewer terms than (9) and is much easier to generate. However, it is not very efficient in parsing time and has issues with numerical stability. Rational expressions require a large number of operations in the parser to generate the signomial form.

In terms of the calculation in the previous section, for this approach we would have $n = 500$, and each of these 500 equations contains all of the 500 h_k we wish to recover. Thus, system is represented with a 500×500 dense matrix.

7) Solution - Indirect IDFT: The previous two formulations were trying to obtain an expression for each sample of the equalized SR. While very natural from designer's perspective, this approach creates very complicated and long expressions with large number of terms. The optimization-based flow offers the possibility to do better by taking a different perspective. This is the central point of our work as we show that both system and circuit level can be designed jointly using existing circuit optimization technology.

This different approach turns the SR formulation into a fitting problem. We give up on the idea of finding exact expression for each sample separately, but try to come up with a set of simpler equations whose solution is exactly the needed SR. In particular the SR samples in our formulation are, from now on, defined as optimization variables and not closed-form expressions, $h_k(\mathbf{p}) \rightarrow h_k$. Furthermore, we wish to avoid numerical problems linked to rational expressions.

To implement these ideas we rewrite (11) and (8) as

$$E(z; \mathbf{p}, \mathbf{h}) = q(z; \mathbf{p}) \sum_{k=0}^{n-1} h_k z^{-k} - p(z; \mathbf{p}) \tag{12}$$

where $p(z; \mathbf{p})$ and $q(z; \mathbf{p})$ are parameterized filters. We can interpret $E(z; \mathbf{p}, \mathbf{h})$ as the fitting error in the $\mathcal{Z}$-domain. With this in mind we impose the following set of constraints in the formulation

$$-tol \leq E(z_j; \mathbf{p}, \mathbf{h}) \leq tol, \quad j = 0, \dots, n-1 \tag{13}$$

where we can perform tradeoff between the fit accuracy and solution time with with $tol \rightarrow 0$.

The obtained formulation (13) is compact and can be generated and parsed quickly. However, it requires long time to solve since each equation contains all the optimization variables[4]. To avoid this we need to perform one more transformation step before passing the formulation to the solver.

---

[4]Due to GP solver algorithm [10] internal operation, a form such as (13), that contains many variables in each expression produces very large matrices when transformed into standard form for solving. While we can see that these matrices are very sparse, during solution each will be decomposed into Cholesky decomposition, and such step does not preserve sparsity. Thus, matrix fill-in becomes an obstacle for efficient operation.

We see that, in (12), it is possible to perform multiplication of $q(z; \mathbf{p})h(z)$, which has the same effect as convolution of coefficients in $q(z; \mathbf{p})$ and $h(z)$. Furthermore, we can represent the fitting error in frequency domain as a sequence of error samples in time, thus arriving to

$$\sum_{k=0}^{n+r-1} e_k(\mathbf{p}, \mathbf{h}) z^{-k} = E(z; \mathbf{p}, \mathbf{h}) =$$
$$= \sum_{k=0}^{n+r-1} \left(\sum_{i=0}^{r-1} q_i(\mathbf{p}) h_{k-i} \right) z^{-k} - p(z; \mathbf{p}) \tag{14}$$

with $deg(z, p(z; \mathbf{p})) \leq n$ which is always satisfied.

The most important point in our derivation of the system level formulation is to note, in (14), that for HSLs $r = deg(z, q(z; \mathbf{p})) << n$. This is always true as we have only few filters (including parasitics) in the signal processing chain, while the number of samples we need to recover is on the order of couple hundreds with even modest oversampling ratio.

To exploit this property we perform IDFT on (14) obtaining

$$e_k(\mathbf{p}, \mathbf{h}) = \sum_{i=0}^{r-1} q_i(\mathbf{p}) h_{k-i} - \frac{1}{n+r-1} \sum_{j=0}^{n+r-1} p(z_j; \mathbf{p}) z_k^j \tag{15}$$
$$k = 0, \dots, n+r-1$$

and impose, similar to (13), set of constraints for time-domain fit

$$-tol \leq e_k(\mathbf{p}, \mathbf{h}) \leq tol, \quad k = 0, \dots, n+r-1 \tag{16}$$

where we constrain, in agreement with (13), that residual error samples of the fitting be bounded within certain $tol \rightarrow 0$.

The formulation in (16) has slightly larger number of equations, $n+r-1$ as compared to n in previous form, but significantly lower number of variables in each, $r << n$. Furthermore, the matrix to describe this formulation is lower triangular. Thus, when fed into the solver, this formulation does not require decomposition as it is already in the form that is appropriate for solving (it has all the properties of the decomposed form). This formulation (16) can be efficiently generated and it performs well with both solver and parser, enabling real-time joint system-and-circuit HSL optimization.[5]

8) A simple example: Suppose that we have a system

$$\sum_{k=0}^{\infty} h_k z^{-k} = \frac{p(z)}{1 - \alpha z^{-1}} \tag{17}$$

which we approximate in 500 samples of h_k

$$\sum_{k=0}^{499} h_k z^{-k} \approx \frac{p(z)}{1 - \alpha z^{-1}} \tag{18}$$

At this point we would be able to perform the sampling using a direct IDFT approach. It is better, as we noted to avoid rational functions and perform the convolution arriving at

$$E(z; \mathbf{p}, \mathbf{h}) = \sum_{k=0}^{500} e_k(\mathbf{p}, \mathbf{h}) z^{-k} =$$
$$= (1 - \alpha z^{-1}) \sum_{k=0}^{499} h_k z^{-k} - p(z) = \tag{19}$$
$$= \sum_{k=0}^{500} (h_k - \alpha h_{k-1}) z^{-k} - p(z)$$

---

[5]Note that if all z_k are chosen such that z_k^* (conjugate) is also used the expression can be calculated quickly as we can simply discard imaginary parts since we know that all the parameters are real.

Performing IDFT in 500 points we finally obtain the time domain fitting formulation

$$e_k(\alpha, \mathbf{h}) = h_k - \alpha h_{k-1} - \frac{1}{500} \sum_{j=0}^{500} p(z_j) z_k^j$$

$$-tol \leq e_k(\alpha, \mathbf{h}) \leq tol, \quad k = 0, \ldots, 500$$

(20)

As we already noted this formulation always has lower-triangular form. The number of side diagonals, below the main diagonal, is equal to the degree of $q(z; \alpha)$. In other words, the number of variables h_k in (20) is only 2 (h_k and h_{k-1}) in each of the 501 constraints e_k, as opposed to 500 in each of 500 constraints that would be obtained from (18) without performing the convolution first.

V. CIRCUIT LEVEL MODEL

With efficient system-level formulation and specified interface variables in place, we next develop circuit level models to complete the problem formulation. Once the circuit topology is chosen, designers have to generate equation-based models for each circuit in a way similar to previous examples of circuit-level equation-based optimization [5], [7], [8].

There are a couple of functions that each circuit model performs in our design flow

- Ensure that topology dictated constraints are met
- Export the interface variables necessary for linking to other blocks and system level
- Export the operating conditions constraints (e.g. dynamic input range)
- Provide an accurate estimate of area and power dissipation of the block
- Ensure that unmodeled effects are not dominant (e.g. constraint on the bandwidth of the parasitic driver filter)

A. Transmit equalizer (TxEq)

For the purpose of demonstrating our design flow we model a tap-sharing $TxEq$ topology [18], since it minimizes the driver parasitics and offers flexibility in programming the equalizer coefficients. Circuit model described here uses the system-level interface variables defined in Table I.

This implementation assumes that the $TxEq$ consists of a number of identical output drivers which are connected to the delay chain through multiplexers in order to encode the tap coefficients. With this implementation it is simple to account for the output impedance of the driver as it is fixed regardless of the value of tap coefficients (within a certain tap coefficient range). However, it introduces some overhead in terms of clocking power due to more complex clock distribution and switching network. To simplify the formulation we define this equalizer block as a collection of the output driver switches. We assume that during normal operation, circuit in Figure 4 behaves as a switch, so biasing constraint equations for minimum output voltage then represent one of the two states of the switch and keep the switch in saturation to prevent the slow recovery (i.e. when the current of the tail transistor M_0 is steered into one of the output branches - i.e. transistor M_1 or M_2).

To ensure the operation in saturation, we export this minimum output voltage to the system ($TxEq$) level. Assuming that tail transistor of the switch is M_0 and differential transistors $M_{1/2}$ we can write biasing constraints in Listing 2.

At the equalizer level we can instantiate the switch as a sub-circuit. Since all the switches are the same, assuming we have N of them we can write

---

Listing 2 Basic biasing constraints for channel driver in TxEq

$M_0.ids = M_1.ids$
$M_0.vds \geq M_0.vdsat$
$M_1.vds \geq M_1.vdsat$
$V_{dd} = M_0.vds + M_1.vgs$
$M_0.vdsat \leq V_{dd} - M_1.vgs$
$V_{dd} - M_1.vgs + M_1.vdsat \leq outVmin$
$M_0.vds + M_1.vds \leq outVmin$

Listing 3 Performance model for TxEq

$swingIN = N(switch.M_0.ids)R_{load}$
$switch.V_{out\_min} \leq V_{dd}OUT - swingIN$

$powerS = N(switch.M_0.ids)V_{dd}$
$C_{in} = 2N\, switch.M_1.G.cap$
$powerD = \alpha f_{Nyquist} C_{in} V_{dd}^2$
$powerTOT = powerD + powerS$

$V_{cm\_out} = V_{dd}OUT - N(switch.M_0.ids)R_{load}/2$

---

where we assume a switching factor α ($\alpha = 0.5$ due to differential operation of the pre-driver). In order to find the average value of the output signal we assumed that the incoming bit sequence is random, so the average is midway between $V_{dd}OUT$ and $V_{dd}OUT - swingIN$. Note that we assume that the channel driver supply $V_{dd}OUT$ can be changed and we use it as a design variable. On the other hand, pre-driver supply voltage is fixed and equal to V_{dd}, and this is reflected in equations in Listings 2, 3 and Figure 4.

B. Receiver equalizer (RxEq)

In this example we model the receive equalizer as a capacitively-degenerated differential peaking amplifier [17], Figure 5. We can determine the transfer function by assuming that the drain impedance is driven by a frequency-selective current source, thus obtaining

$$\frac{v_{out}}{v_{in}}(s) = \frac{g_m R_D}{1 + g_m R_S} \frac{1 + sR_S C_S}{(1 + s\frac{R_S C_S}{1 + g_m R_S})(1 + sR_D C_D)}$$

(21)

where g_m is the transconductance of $M_{1/2}$ and R_S, R_D and C_S, C_D are equivalent source and drain resistance and capacitance, respectively.

To ensure proper biasing of the circuit once the system is put together we must pass the available input and output signal dynamic range to the upper levels, which depends on transistor sizing through the constraints for saturation operating conditions. In order to limit the nonlinear distortion, we assume that only a certain percentage of the tail (bias) current can be steered into one of the input transistors. To do this, we create three instances of the same input transistor (i.e. M_1) and assume different biasing currents, Listing 4, where
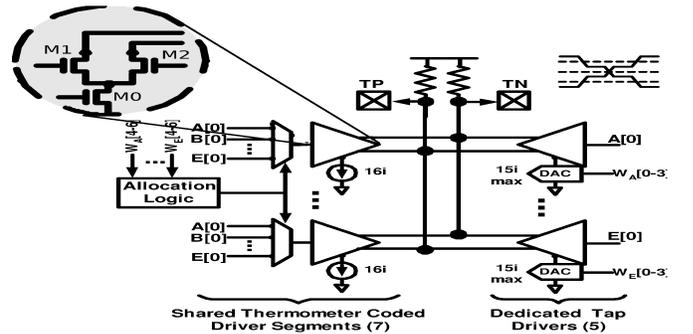

Fig. 4. Tap-sharing equalizer topology with differential switch as a basic building block

Listing 4 Basic biasing constraints for RxEq

$M_{1\_tail}.vdsat + M_1\_hi.vgs \le Vin\_min$
$M_{1\_tail}.vdsat + M_1\_lo.vgs \le Vin\_max$

$V_{dd} - (M_1\_hi.ids)R_{load} - M_1\_hi.vdsat + M_1\_hi.vgs \ge Vin\_max$
$M_1\_hi.vgs - M_1\_lo.vgs > Vin\_max - Vin\_min$

$Vin\_max > Vin\_cm > Vin\_min$

$I_{HI} = 1.8M_{1\_tail}.ids$
$I_{LO} = 0.2M_{1\_tail}.ids$
$M_1\_hi.ids \le I_{HI}$
$M_1\_lo.ids = 2M_1.ids - M_1\_hi.ids$
$statP = 2V_{dd}M_{1\_tail}.ids$

we used instances M_1_hi and M_1_lo to represent input transistor (M_1) in different operating conditions when signal at its gate achieves maximum and minimum value. We assumed that we can, at most, switch between 20% and 80% of the total bias current, which is $M_{1\_tail}.ids + M_{2\_tail}.ids = 2M_{1\_tail}.ids = 2M_1.ids$.

Finally, we discretize this analog filter transfer function to fit in our system level formulation, using Tustin's transform[6]

$$\frac{v_{out}}{v_{in}}(z) = \frac{g_m}{C_D}\frac{T_s}{2}\frac{1 + \frac{\omega_z T_s}{2}}{(1 + \frac{\omega_{pd}T_s}{2})(1 + \frac{\omega_{ps}T_s}{2})}h(z)$$

$$h(z) = (1 + z^{-1})\frac{1 - \frac{1 - \frac{\omega_z T_s}{2}}{1 + \frac{\omega_z T_s}{2}}z^{-1}}{(1 - \frac{1 - \frac{\omega_{pd}T_s}{2}}{1 + \frac{\omega_{pd}T_s}{2}}z^{-1})(1 - \frac{1 - \frac{\omega_{ps}T_s}{2}}{1 + \frac{\omega_{ps}T_s}{2}}z^{-1})} \tag{22}$$

and time constants of the model can be defined as

$$\tau_S = R_S C_S, \quad \tau_D = R_D C_D, \quad \omega_z = 1/\tau_S$$
$$\omega_{ps} = (1 + g_m R_S)/\tau_S, \quad \omega_{pd} = 1/\tau_D \tag{23}$$

We take into account the parasitic capacitance in the drain of $M_{1/2}$ in C_D along with C_p from Figure 5. We can finally link the circuit parameters to the system level parameters we defined in Table II

$$scale = \frac{g_m}{C_D}\frac{T_s}{2}\frac{1 + \frac{\omega_z T_s}{2}}{(1 + \frac{\omega_{pd}T_s}{2})(1 + \frac{\omega_{ps}T_s}{2})}$$

$$zRX_1 = \frac{1 - \frac{\omega_z T_s}{2}}{1 + \frac{\omega_z T_s}{2}} \tag{24}$$

$$dpRX_1 = \frac{1 - \frac{\omega_{pd}T_s}{2}}{1 + \frac{\omega_{pd}T_s}{2}}, \quad spRX_1 = \frac{1 - \frac{\omega_{ps}T_s}{2}}{1 + \frac{\omega_{ps}T_s}{2}}.$$

C. Predriver/Slicer/DFE

The predriver, slicer and DFE are currently included as place-holders, with defined interface variables. For example, predriver

---

[6]We choose this transformation due to the mapping between analog and digital domains that can be expressed in signomial form $s = \frac{2}{T_s}\frac{z-1}{z+1}$.
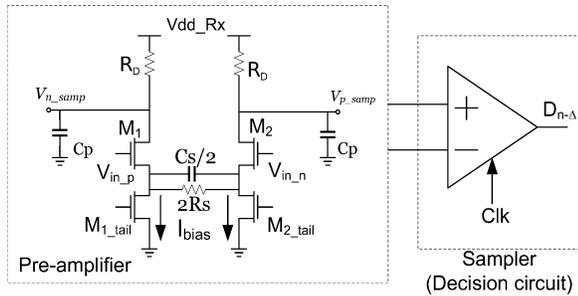

Fig. 5. Capacitively peaked receiver equalizer

Listing 5 Metacode for system and circuit linking

```
% instantiation
system = subCkt(system_description);
txPredriver = subCkt(txPre);
txEQ = subCkt(tx_equalizer);
rxEQ = subCkt(rx_equalizer);
...
% system variable linking
system.swingIN = txEQ.swingIN;
system.zRX = rxEQ.zRX;
...
% operating conditions
rxEQ.max_Vin ≥ txEQ.comm_mode + system.DR;
xEQ.min_Vin ≥ txEQ.comm_mode - system.DR;
...
% performance
2system.d_min[k] ≥ slicer.sensitivity + (slicer.sigma)(Q(1e-15));
% objective
min: txPredriver.power + txEQ.power + rxEQ.power + slicer.power;
```

power is estimated according to the driver input capacitance and fan-out sizing rules that depend on the link data-rate. Similarly, the slicer is currently abstracted through its input biasing requirements and system-level sensitivity and noise variables. While modeling of any link circuit requires careful consideration, we do not foresee significant difficulties in including more accurate circuit models for these blocks in our optimization flow.

VI. PUTTING EVERYTHING TOGETHER

With all the modules in place, and defined interface variables, the last step is to formally link all of them. In our example this is done by instantiating all the necessary models as sub-blocks (in this way even the system level formulation becomes a sub-block), Listing 5. Then we simply impose equality constraints between appropriate interface variables, which is made easy by the signomial-based formulation. We also link operating conditions of each circuit to the overall system settings at this level. Finally, the sampling phase has to be chosen at which we specify the BER constraint (1).

In a sense, we can think of our system level formulation as a specification setting layer. For any change in transmit equalization it will readjust the input dynamic range into the receiver equalizer as well as recalculate the eye aperture thus changing the specifications of the receiver, and vice versa. We can see this, for example, in Listing 5 in sections on *operating conditions* where input dynamic range of the peaking amplifier is adjusted according to system level estimate of the dynamic range of the signal at its input (4). This is exactly where the iterative loop of the traditional design flow is replaced.
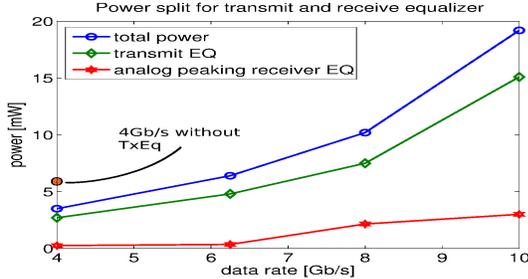
A. Results

To test our formulation we try to design an HSL system communicating over a 32 inch channel with transfer function given in Fig. 3(b). Our objective is power minimization while requiring that $d_{min}[k] \ge 50mV$. In our circuit formulations we used signomial and posynomial transistor models extracted from predictive $90nm$ technology models.

The trade-off between link power and sampling phase is shown in Fig. 6(a) indicating a significant dependence on the sampling position and offering the possibilities for efficient exploration of equalization and CDR interaction.

By running the formulation at multiple data rates and picking the instances with best power results we can see how the system power scales with data rate, Fig. 6(b). We run optimizations for $4Gb/s$,
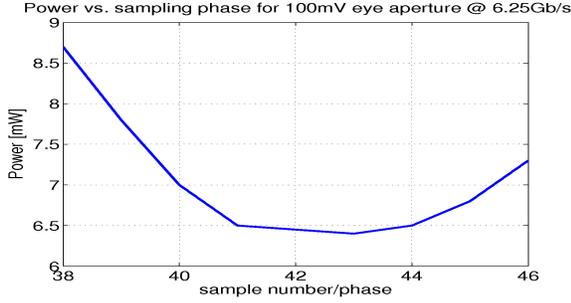
(a) Power vs. sampling phase for 6.25Gb/s, no DFE

(b) Power vs. symbol-rate

Fig. 6.   Dependency of power on symbol rate and sampling phase

$6.25Gb/s$, $8Gb/s$ and $10Gb/s$, Table IV. The results suggest that $d_{min}[k] \geq 50mV$ constraint is achievable at $8Gb/s$ only with 1-tap DFE and at $10Gb/s$ we could only achieve $d_{min}[k] \geq 35mV$ even with 4-tap DFE. Here, the DFE assumption is introduced only at the system level, by omitting certain number of post-cursors from the $d_{min}[k]$ equation (2). This means that the overall power numbers for $8Gb/s$ and $10Gb/s$ are optimistic. On the receiver side, the analog peaking amplifier consumes around $5\% - 10\%$ of power at lower speeds and between $20\% - 30\%$ as we keep increasing the speed. It is interesting to see that, for the given link topology, no solution actually had significant peaking in the $RxEq$. In all our design attempts, the best solution was using wideband analog receiver amplifier in $RxEq$ with the bandwidth always above the Nyquist frequency.

Since this choice defies conventional wisdom, we further investigate this result in more detail. In the first column of Table IV we show the results of an optimization-run without any transmit side equalization, where the receive peaking amplifier takes on the role of the equalizer. Note that in this situation, $RxEq$ is forced to provide high-frequency gain under large input dynamic range, which requires a lot of power. Since in capacitively peaked amplifier, peaking is achieved by lowering the DC gain and higher overall system power than in the case where $TxEq$ and receive amplifier are used (second column of Table IV), Figure 6(b). In the case with no $TxEq$ the system compensates for the decrease of $RxEq$ DC gain by increasing

TABLE IV
SOME OPTIMIZATION RESULTS

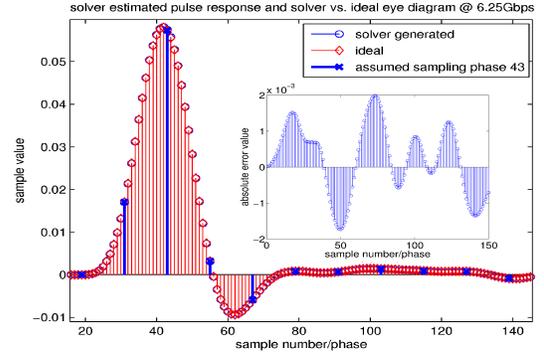| rate [Gb/s]	4	4	6.25	8	10
swingIN [mV]	200	100	200	280	330
aFR	none	$\begin{bmatrix} 1 \\ -.42 \end{bmatrix}$	$\begin{bmatrix} 1 \\ -.52 \end{bmatrix}$	$\begin{bmatrix} 1 \\ -.42 \\ -.05 \end{bmatrix}$	$\begin{bmatrix} 1 \\ -.43 \\ -.15 \end{bmatrix}$
RxEq gain	1.1	3.5	2.7	2.1	3.3
$\omega_z/(2\pi)$ [GHz]	0.8	2.4	4.7	2.2	3.4
$\omega_{dp}/(2\pi)$ [GHz]	2.1	3.4	5.7	3.6	3.6
$\omega_{sp}/(2\pi)$ [GHz]	2.6	5.4	5.8	10	3.3

Fig. 7.   Accuracy of calculated symbol response using formulation (16) against Matlab calculation

the transmit side swing. This interaction shows that our system level formulation does in fact manage to express tight inter-dependencies between system blocks that optimization engine can utilize. For a designer who is just focused on the block level, these interactions are hard to capture.

Looking further in the table, we can observe that from $4Gb/s$ to $6.25Gb/s$ optimizer chooses to apply more aggressive equalization at the transmitter. This is expected, as the channel has stronger postcursors at $6.25Gb/s$ and more equalization is needed. However, this trend does not continue through $8Gb/s$ and $10Gb/s$ cases, since at these data rates the channel attenuation is so high that only DFE can meet the BER requirements. Since in our setup, the optimizer does not see any penalty to utilizing DFE (no circuit model is supplied, thus no power increase is observed) it performs most of the equalization with DFE. The $TxEq$ is still utilized to control the peak-to-average ratio of the signal entering the receive amplifier due to our constraint on input dynamic range of this circuit in order to preserve linearity. We see that transmit equalizer can be used for equalization purposes in order to achieve better eye aperture, but also as a means to control operating conditions of the receive amplifier. In short, our results indicate that interaction between different methods of equalization could be studied through the developed framework.

B. Verification

Finally, we check the accuracy of our formulation method when compared to the standard tools used in HSL design.

In Fig. 7 we compare symbol response for one of the cases discussed (optimal sampling phase for $6.25Gb/s$) obtained in our optimization formulation against appropriate Matlab calculated response for the same equalization settings. We see exceptional matching between these waveforms with less than $2mV$ of absolute error e_k. This can be further improved by tightening optimizer parameters with solution time penalty as we noted in reference to (13).

Finally, we run Spectre simulations for the design obtained for $6.25Gb/s$ presented in Table IV to confirm proper biasing and performance of the $TxEq$ and $RxEq$ circuit blocks. We found that transistor biasing was done properly to the $10 - 30\%$ accuracy in terms of saturation voltages. $TxEq$ and $RxEq$ parameters (output swing, gains, poles and zeros) were all matched to within 10%[7].

---

[7]The AC pole-zero analysis in Spectre revealed that source impedance-induced pole and zero of the RxEq require better modeling due to the neglected positive feedback through output conductance g_{ds} of the input transistors $M_{1/2}$. With appropriate correction our results match the simulation well.

C. Computational efficiency

We tested our design flow by running design optimizations for the channel in Figure 3 at $4Gb/s$, $6.25Gb/s$, $8Gb/s$ and $10Gb/s$. The system formulation dominated the size of the whole formulation in terms of the number of different optimization variables and optimization constraints. Thus the parsing and solution time were mostly due to the system level formulation. In general, our problems had between 1500 and 1700 variables and between 4500 and 5500 constraints, depending on the actual number of significant samples in the SR, which changes with data rate.

In terms of run time, the developed flow is very efficient. Our optimization runs were executed on a laptop with $1.60GHz$ Pentium M processor and $2GB$ of memory running Debian Linux. The parsing time varied between 15 minutes and 1 hour, while the solver time was between 1 hour and 3 hours. The overall solution time was below 4 hours for any of the configurations.

The developed formulation is a very efficient tool for design space exploration. It is well-suited for design of such complicated systems as HSLs over multiple channels and process corners. With appropriate formulation and circuit model adjustments it could be applied to any mixed signal front-end design.

VII. CONCLUSIONS

Design of power-constrained analog/mixed-signal front-ends, especially radios and high-speed links requires careful balance of power allocation among the circuits in the signal chain. In traditional iterative design approach, the system performance modeling and circuit design are rather disjoint or at best connected in an ad-hoc manner. By casting the high-speed link performance model as a symbolic signal transformation problem, with filters parameterized by circuit performance properties, we were able to establish an efficient problem formulation for joint system-and-circuit optimization. This efficient formulation improves the tractability of the optimization by several orders of magnitude and enables real-time optimization. We illustrated the potential of this framework on a simplified high-speed link example, showing the potential of the new formulation. Although quite simplistic, this example allowed us to address for the first time the possible trade-offs between transmit and receive equalizers in a unified manner, showing the mechanism through which they interact with channel loss and other circuit constraints like input and output dynamic range. To the best of our knowledge, this is the first work to introduce transistor level circuit models in system level design of an HSL by means of optimization.

VIII. ACKNOWLEDGMENTS

REFERENCES

[1] B. Casper, M. Haycock, and R. Mooney, "An accurate and efficient analysis method for multi-gb/s chip-to-chip signaling schemes," *Symposium on VLSI Circuits Digest of Technical Papers, 2002.*, no. SN -, pp. 54–57, 2002.

[2] V. Stojanovic and M. Horowitz, "Modeling and analysis of high-speed links," in *IEEE Custom Integrated Circuits Conference*, 2003.

[3] P. Hanumolu, B. Casper, R. Mooney, G.-Y. Wei, and U.-K. Moon, "Analysis of pll clock jitter in high-speed serial links," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 50, no. 11 SN - 1057-7130, pp. 879–886, 2003.

[4] H. Hatamkhani and C. Ken Yang, "A study of the optimal data rate for minimum power of i/os," *IEEE Transactions on Circuits and Systems II*, vol. 53, no. 11 SN - 1057-7130, pp. 1230–1234, 2006.

[5] M. Hershenson, S. Boyd, and T. Lee, "Optimal design of a cmos op-amp via geometric programming," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 20, no. 1 SN - 0278-0070, pp. 1–21, 2001.

[6] P. Mandal and V. Visvanathan, "Cmos op-amp sizing using a geometric programming formulation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 20, no. 1, pp. 22–38, 2001.

[7] M. del Mar Hershenson, "Design of pipeline analog-to-digital converters via geometric programming," in *ICCAD '02: Proceedings of the 2002 IEEE/ACM international conference on Computer-aided design*, 2002, pp. 317–324.

[8] D. Colleran, C. Portmann, A. Hassibi, C. Crusius, S. Mohan, S. Boyd, T. Lee, and M. del Mar Hershenson, "Optimization of phase-locked loop circuits via geometric programming," *Proceedings of the IEEE Custom Integrated Circuits Conference, 2003.*, no. SN -, pp. 377–380, 2003.

[9] X. Li, J. Wang, L. Pileggi, T.-S. Chen, and W. Chiang, "Performance-centering optimization for system-level analog design exploration," in *IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, 2005, pp. 422–429.

[10] S. P. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, UK: Cambridge University Press, 2004.

[11] C. Maranas and C. Floudas, "Global optimization in generalized geometric programming," pp. 351–370, 1997.

[12] J. Kim, J. Lee, L. Vandenberghe, and C. Yang, "Techniques for improving the accuracy of geometric-programming based analog circuit design optimization," *IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, no. SN - 1092-3152, pp. 863–870, 2004.

[13] W. Daems, G. Gielen, and W. Sansen, "Simulation-based automatic generation of signomial and posynomial performance models for analog integrated circuit sizing," *IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, no. SN -, pp. 70–74, 2001.

[14] V. Stojanovic, A. Amirkhany, and M. Horowitz, "Optimal linear pre-coding with theoretical and practical data rates in high-speed serial-link backplane communication," *IEEE International Conference on Communications*, vol. 5, no. SN -, pp. 2799–2806 Vol.5, 2004.

[15] J. Ren and M. Greenstreet, "A unified optimization framework for equalization filter synthesis," in *DAC '05: Proceedings of the 42nd annual conference on Design automation*, 2005, pp. 638–643.

[16] D. Bertsimas and J. N. Tsitsiklis, *Introduction to Linear Programming*. Athena Scientific, 1997.

[17] R. Farjad-Rad, N. Hiok-Taiq, M. Edward Lee, R. Senthinathan, W. Dally, A. Nguyen, R. Rathi, J. Poulton, J. Edmondson, J. Tran, and H. Yazdanmehr, "0.622-8.0 gbps 150 mw serial io macrocell with fully flexible preemphasis and equalization," *Symposium on VLSI Circuits, 2003. Digest of Technical Papers. 2003*, no. SN -, pp. 63–66, 2003.

[18] J. Zerbe, C. Werner, V. Stojanovic, F. Chen, J. Wei, G. Tsang, D. Kim, W. Stonecypher, A. Ho, T. Thrush, R. Kollipara, M. Horowitz, and K. Donnelly, "Equalization and clock recovery for a 2.5-10-gb/s 2-pam/4-pam backplane transceiver cell," *IEEE Journal of Solid-State Circuits*, vol. 38, no. 12 SN - 0018-9200, pp. 2121–2130, 2003.