Mixed Signal Design Space Exploration through Analog Platforms

F. De Bernardinis[†§]
fdb@eecs.berkeley.edu,
f.debernardinis@ing.unipi.it

P. Nuzzo[§]
pierluigi.nuzzo@iet.unipi.it

A. Sangiovanni Vincentelli[†]
alberto@eecs.berkeley.edu

[†]Department of Electrical Engineering and Computer Science
University of California, Berkeley

[§]Dipartimento di Ingegneria dell'Informazione
Università di Pisa, Italy

ABSTRACT

We propose a hierarchical mixed signal design methodology based on the principles of Platform-Based Design (PBD). The methodology is a meet-in-the-middle approach where design components are modeled bottom-up at various abstraction levels and performance constraints are mapped top-down to select among the available components the ones that best meet the constraints. The design methodology can seamlessly operate on both analog and digital designs, thus dealing with mixed signal designs in a consistent way. We demonstrate the effectiveness of the approach optimizing an 80 MS/s 14 bit pipelined Analog-to-Digital Converter (ADC) including digital calibration, yielding 64% power reduction compared to the original hand optimized design.

Categories and Subject Descriptors

B.7 [**Integrated Circuits**]: Design Aids

General Terms

Algorithms

1. INTRODUCTION

The challenges of mixed signal design involve all levels of abstraction, from specification to physical implementation. We address here the problem of mixed-signal *system-level* design that has received scant attention from the design community. In particular, we are interested in design space exploration and analog/digital tradeoff analysis. Motivated by the evolution of Platform-based Design (PBD) methodologies for the digital domain, we develop a design methodology (Analog Platform-based Design (APBD)) for analog design that shares some of its principles so that a coherent platform-based mixed signal methodology can be constructed. APBD is based on abstraction mappings to yield high-level models of analog components with performance models that can be made accurate, a requirement for high-level design space exploration to provide solutions that can indeed yield the estimated performance. Performance models are at the heart of APBD. In the literature, several attempts exist for approximating circuit performances, mostly based on regression schemes, as in [1, 2]. In [3] an efficient approximation methodology based on polytopes is presented based on linear interpolation of simulated performances. We propose a performance modeling scheme that makes no assumption on the performance space and relies on statistical simulation schemes. Even if the characterization process is more expensive, a *conservative* model is generated in the sense that it minimizes errors by tight control of extrapolation of simulated performances. Design exploration is performed by composing high-level performance and behavioral models in a variety of configurations and optimizing the parameters to minimize an appropriate cost function. Finally, since APBD is consistent with digital PBD, analog/digital trade-offs can be performed in a coherent framework.

2. ANALOG PLATFORMS

The PBD paradigm consists of a *meet-in-the-middle* design approach that provides an effective answer to several engineering and economical issues [4]. A platform consists of a *library* of components, interconnects and rules that define legal composition of the components. At each level of abstraction, a design is represented by a *platform instance*, a legal composition of library elements. PBD consists of two separate phases: bottom-up library development and characterization, and top-down mapping. Developing platforms requires characterizing bottom-up both behavior and performances of platform elements and providing the necessary tools to map higher level specifications into lower level constraints. The library also provides composition rules to connect components, so that correct platform instances can be isolated from unfeasible combinations. The top-down phase consists of mapping a design (top level platform instance) to a lower level platform instance so that a set of constraints is satisfied and some cost function is minimized.

2.1 Definition

The platform abstraction process can be extended to analog components in a very natural way. Deriving behavioral and performance models, however, is more involved due to the tight dependency of analog components on device physics that requires the use of continuous mathematics for modeling the relations among design variables. Formally, an Analog Platform (AP) consists of a set of components, each decorated with:

- a set of input variables $u \in \mathcal{U}$, a set of output (performance) variables $y \in \mathcal{Y}$, a set of "internal" variables (including state variables) $x \in \mathcal{X}$, a set of configuration parameters $\kappa \in \mathcal{K}$; some parameters take values in a continuous space, some take values in a discrete set, for example when they encode the selection of a particular alternative;
- a *behavioral model* that expresses the behavior of the component represented implicitly as $\mathcal{F}(u, y, x, \kappa) = 0$, where $\mathcal{F}(\cdot)$, may include integro-differential components; in general, this set determines uniquely x and y given u and κ. Note that the variables considered here can be function of time and that the functional $\mathcal{F}$ includes constraints on the set of variables (for example, the initial conditions on the state variables).
- a *feasible performance model.* Let $\phi_y(u, \kappa)$ denote the map that computes the performance y corresponding to a particular value of u and κ by solving the behavioral model. The set of feasible analog performance (such as gain, distortion, power), is the set described by the relation $\mathcal{P}(y(u)) = 1 \Leftrightarrow \exists \kappa', y(u) = \phi_y(\kappa', u)$.
- *validity laws* $\mathcal{L}(u, y, x, \kappa) \leq 0$ i.e., constraints (or assumptions) on the variables and parameters of the component that define the range of the variables for which the behavioral and performance models are valid.

Note that there is no real need to define the feasible performance model since the necessary information is all contained in the behavioral model. We prefer to keep them separate because of the use we make of them in explaining our approach.

At the circuit level of abstraction, the behavioral models are the circuit equations with x being the voltages, currents and charges, y being a subset of x and/or a function of x and κ when they express performance figures such as power or gain. To compute performance models, we need to solve the behavioral models that implies solving ordinary differential equations, a time consuming task. In the past, methods to approximate the relation between y and κ (the design variables) with an explicit function were proposed. In general, to compute this approximation, a number of evaluations of the behavioral model for a number of parameters κ is performed (by simulation, for example) and then an interpolation or approximation scheme is used to derive the approximation to the map ϕ_y. We see in Section 3 how to compute an approximation to the feasible performance set directly.

Example — Considering an OTA for an arbitrary application, we can start building a platform from the circuit level by defining:

- $\mathcal{U}$ as the set of all possible input voltages $V_{in}(t)$ s.t. $|V_{in}| < 100$ mV and bandwidth $V_{in} < 3$ MHz; $\mathcal{Y}$ as the space of vectors $\{V_{out}(t), \text{gain}, \text{IIP3}, r_{out}\}$ (IIP3 is the third order intermodulation intercept point referred to the input, r_{out} is the output resistance) $\mathcal{X}$ the set of all internal current and voltages, and $\mathcal{K}$ the set of transistor sizings.
- for a transistor level component, the behavioral model $\mathcal{F}$ consists of the solution of the circuit equations, e.g. through a circuit simulator.
- $\phi_y(u, \kappa)$ as the set of all possible y;
- validity laws $\mathcal{L}$ are obtained from Kirchoff laws when composing individual transistors and other constraints, e.g. maximum power ratings of breakdown voltages.

We can build a higher level (level 1) OpAmp platform where:

- $\mathcal{U}^1$ is the same, $\mathcal{Y}^1$ is the output voltage of the OpAmp, $\mathcal{X}$ is empty, $\mathcal{K}^1$ consists of possible {gain, IIP3, r_{out}} triples (thus it is a projection of $\mathcal{Y}^0$);
- $\mathcal{F}^1$ can be expressed in explicit form,

$$\begin{cases} y_1(t) = h(t) \otimes (a_1 \cdot u(t) + a_3 \cdot u(t)^3) + \text{noise} \\ y_2 = a_1; y_3 = \sqrt{\frac{4}{3} \cdot \frac{a_1}{a_3}} \end{cases}$$

- ϕ_y is the set of possible y;
- there are no validity constraints, $\mathcal{L} < 0$ always. ■

When a platform instance is considered, we have to compose the models of the components to obtain the corresponding models for the instance. The platform instance is then characterized by

- a set of internal variables of the platform $\xi = [\xi_1, \xi_2, ..., \xi_n] \in \Xi$,
- a set of inputs of the platform, $h \in \mathcal{H}$
- a set of performances $\upsilon \in \Upsilon$,
- a set of parameters $\zeta \in \mathcal{Z}$.

The variable names are different from the names used to denote the variables of the components to stress that there may be situations where some of the component variables change roles (for example, an input variable of one component may become an internal variable; a new parameter can be identified in the platform instance that is not visible or useful at the component level). To compose the models, we have to include in the platform the composition rules. The *legal compositions* are characterized by the interconnect equations that specify which variables are shared when composing components and by constraints that define when the composition is indeed possible. These constraints may involve range of variables as well as nonlinear relations among variables. Formally, a connection is establishing a pairwise equality between internal variables for example $\xi_i = \xi_j$, inputs and performance; we denote the set of *interconnect relations* with $c(h, \xi, \zeta, \kappa) = 0$ that are in general a set of linear equalities. The *composition constraints* are denoted by $\mathcal{L}(h, \xi, \upsilon, \zeta) \leq 0$, that are in general, non linear inequalities. Note that in the platform instance all internal variables of the components are present as well as all input variables. In addition, there is no internal or input variable of the platform instance that is not an internal or input variable of one of the components. The behavioral model of the platform instance is the union of all behavioral models of the components conjoined with the interconnect relations. The validity laws are the conjunction of the validity laws of the components and of the composition constraints. The feasible performance model may be defined anew on the platform instance but it may also be obtained by composition of the performance models of the components. There is an important and interesting case when the composition may be done considering only the feasible performance models of the components obtained by appropriate approximation techniques. In this case, the composition constraints assume the semantics of defining when the performance models may be composed. In this case, an architectural exploration step consisting of forming different platform instances out of the component library and evaluating them, can be performed very quickly albeit possibly with restrictions on the space of the considered instances caused by the composition constraints.

Example — We can build a level 2 platform platform consisting of an OpAmp (OA) and a unity gain buffer following it (UB, the reader can easily find a proper definition for it), then we can define a higher level OpAmp platform component so that:

- $\xi_1 = V_{in}^{OA}$, $\xi_2 = V_{out}^{OA}$, $\xi_3 = V_{in}^{UB}$, $\xi_4 = V_{in}^{UB}$ and connect them in series specifying $\xi_2 = \xi_3$;
- h connected to ξ_1 is the set of input voltages $V_{in}(t)$;
- Υ is the space of $\upsilon_1(t)$, the cascade response in time, $\upsilon_2 = gain$, $\upsilon_3 = IIP3$. In this case υ_2 immediately equals y_2^{OA}, while υ_3 is a non linear function of y^{OA} and y^{UB};
- $\mathcal{Z}$ consists of all parameters specifying a platform instance, in this case we may have $\mathcal{Z} = \mathcal{Y}_{OA} \cup \mathcal{Y}_{UB}$.
- a platform instance composability law $\mathcal{L}$ requires that the load impedance $Z_L > 100 r_{out}$ both at the output of the OpAmp and the unity buffer.

■

2.2 Mixed-Signal Design Flow with Platforms

The essence of platform-based design is building a set of abstractions that facilitate the design of complex systems by a successive refinement/abstraction process. The abstraction takes place when an existing set of components forming a platform at a given level of abstraction is elevated to a higher level by building appropriate behavioral and performance models together with the appropriate validity laws. This process can take either components at a level of abstraction and abstract each of them, or abstract a set of platform instances. Since both platform instances and platform components are described at the same level of abstraction the process is essentially the same. What changes is the exploration approach. On the other side of the coin, the top-down phase progresses through refinement. Design goals are captured as constraints and cost function. At the highest level of abstraction, the constraints are intersected with the feasible performance set to identify the set of achievable performance that satisfy design constraints. The cost function is then optimized with respect to the parameters of the platform instances at the highest level of abstraction ensuring they lie in the intersection of the constraint and the feasible performance set. This constrained optimization problem yields a point in the feasible performance space and in the parameter space for the platform instances at the highest level of abstraction. Using the inverse of the abstraction map ϕ_y, these points are mapped back at a lower level of abstraction where the process is repeated to yield a new point in the achievable performance set and in the parameter space until we reach a level where the circuit diagrams and even a layout is available. If the abstraction map is a conservative map, then every time we map down, we always find a consistent solution that can be achieved. Hence the design process can be shortened considerably. The crux of the matter is how many feasible points are not considered because of the conservative approximation. Thus the usual design speed versus design quality trade-off has to be explored.

In mathematical terms, the bottom-up phase consists of defining an abstraction ψ^l that maps the inputs, performance, internal variables, parameters, behavioral and performance models, and validity laws of a component or platform instance at level l into the corresponding objects at the next level $(l+1)$. The map is *conservative* if all feasible performance vectors $\hat{y}^{l+1}$ correspond to feasible performance vectors $\hat{y}^l$. Note that if approximations are involved in defining the models and the maps, this is not necessarily true, i.e., abstraction maps may be non conservative. In other words, a feasible performance vector at level $l+1$ may not correspond to a feasible one at level l.

The top-down phase then proceeds formulating a top-level design problem as an optimization problem with a cost function $\mathcal{C}(y_{top})$ and a set of constraints defined in the $\mathcal{Y}_{top}$ space, $g_{top}(y_{top}) \leq 0$ that identifies a feasible set in $\mathcal{Y}_{top}$. The complete optimization problem has to include the set $\hat{\mathcal{Y}}_{top}$ that defines the set of achievable performance at the top level. The intersection of the two sets define the feasible set for the optimization process. The result of the process is a ${y_{top}}^{opt}$. Then the process is to map back the selected point to the lower levels of the hierarchy. If the abstractions are conservative, the top-down process is straightforward. Otherwise, at each level of the hierarchy, we have to verify using the performance models, the behavioral models and the validity laws. In some cases, a better design may be obtained by introducing in the top-down phase cost functions and constraints that are defined only at a particular abstraction level. In this case, the space of achievable performances intersected with this new set of constraints defines the search space for the optimization process. At times, it is more convenient to project down the cost function and the constraints of the higher-level abstraction to the next level down. In this case, then the search space is the result of the intersection of three sets in the performance space and the cost function is a combination of the projected cost function and the one defined at this level.

The peculiarity of a platform approach to mixed signal design resides in the accurate performance model constraints $\mathcal{P}$ that propagate to the top-level architecture related constraints. For example, a platform stack can be built where multiple analog implementation architectures are presented at a common level of abstraction together with digital enhancement platforms (possibly including several algorithms and hardware architectures), each component being annotated with feasible performance spaces. Solving the system design problem at the top level where the platforms contain both analog and digital components, allows selecting optimal platform instances in terms of analog and digital solutions, comparing how different digital solutions interact with different analog topologies and finally selecting the best tradeoff.

The final verification step is also greatly simplified by the platform approach since, at the end, models and performances used in the top-down phase were obtained with a bottom-up scheme. Therefore, a consistency check of models, performances and composition effects is all that is required at a hierarchical level, followed by more costly, low-level simulations that check for possible important effects that were neglected when characterizing the platform.

3. BUILDING PERFORMANCE MODELS

An important part of the methodology is obtaining performance models. We already mentioned that we need to approximate the set $\hat{\mathcal{Y}}$ explicitly eliminating the dependence on the internal variables x. To do so a simulation-based approach is proposed.

3.1 Performance Model Approximation

In general terms, simulation maps a configuration set (typically connected) $\mathcal{K}$ into a performance set in $\mathcal{Y}$, thus establishing a relation among points belonging to the mapped set. Classic regression schemes provides an efficient approximation to the mapping function $\phi(\cdot)$, however our approach requires dealing with performance data in two different ways. The first one, referred to as performance model $\mathcal{P}$, allows discriminating between points in $\hat{\mathcal{Y}}$ and points in $\mathcal{Y} \backslash \hat{\mathcal{Y}}$. A second one, $\mu(\cdot) = \phi^{-1}(\cdot)$, implementing the inverse mapping from $\hat{\mathcal{Y}}$ into $\mathcal{K}$, used to map down from a higher-level platform layer to a lower one. However, fundamental issues (i.e. $\phi(\cdot)$ being an invertible function) and accuracy issues (a regression from $\mathbb{R}^m$ into $\mathbb{R}^n$) suggest a table-lookup implementation for $\mu(\cdot)$, possibly followed by a local optimization phase to improve mapping. Therefore, we will mainly focus on basic performance models $\mathcal{P}$.

The set $\hat{\mathcal{Y}} \subset \mathcal{Y}$ defines a relation in $\mathcal{Y}$ denoted with $\mathcal{P}$. We use Support Vector Machines (SVMs) as a way of approximating the performance relation $\mathcal{P}$ [5]. SVMs provide approximating functions of the form

$$f(x) = \operatorname{sign}(\sum_i \alpha_i e^{-\gamma |x - x_i|^2} - \rho) \tag{1}$$

where x is the vector to be classified, x_i are observed vectors, α_is are weighting multipliers, ρ is a biasing constant and γ is a parameter controlling the fit of the approximation. More specifically, SVMs exploit mapping to Hilbert spaces so that hyperplanes can be exploited to perform classification. Mapping to high dimensional spaces is achieved through *kernel* functions, so that a kernel $k(\kappa, \cdot)$ is associated at each point κ. Since the only general assumption we can make on $\phi(\cdot)$ is continuity and on $\mathcal{K}$ is connectivity [1], we can only deduce that $\hat{\mathcal{Y}}$ is connected as well. Therefore, the radial basis function Gaussian kernel is chosen, $k(\kappa, \kappa') = e^{-\gamma \cdot \|\kappa - \kappa'\|^2}$, where γ is a parameter of the kernel and controls the "width" of the kernel function around κ. We resort to a particular formulation of SVMs known as one-class SVM where an optimal hyperplane is determined to separate data from the origin. The optimal hyperplane be computed very efficiently through a quadratic problem, as detailed in [6].

3.2 Optimizing the approximation process

Sampling schemes for approximating unknown functions are exponentially dependent on the size of the function support. In the case of circuit, none but very simple circuits could be realistically characterized in this way. Fortunately, there is no need to sample the entire space $\mathcal{K}$ since we can use additional information obtained from design considerations to exclude parts of the parameter space. The set of "interesting" parameters is delimited by a set of constraints of two types:

[1]more in general, a union of a finite number of connected sets

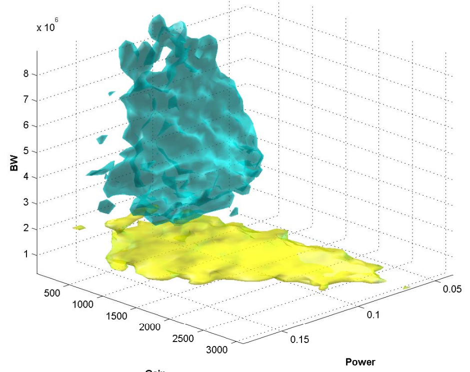

Figure 1: Performance model projection into Power, Gain, Bandwidth subspace for the two OTA topologies considered in the case study. The dark region refers to the telescopic topology. Different Gain/BW tradeoffs are clearly visible across topologies.

- *topological* — constraints derived from the use of particular circuit structures, such as two stacked transistor sharing the same current or a set of V_{DS} summing to zero;
- *physical* — constraints induced by device physics, such as V_{GS}-V_{DS} relation to enforce saturation or g_m-I_D relations;

Additional constraints can be added as designers' understanding of circuit improves. The more constraints we add, the smaller the interesting configuration space $\mathcal{K}$. However, if a constraint is tight, i.e., it either defines lower dimensional manifolds for example when the constraint is an equality, or the measure of the manifold is small, the more likely it is to introduce some bias in the sampling mechanism because of the difficulty in selecting points in these manifolds. To eliminate this ill-conditioning effect, we "relax" these constraints to include a larger set of interesting parameters. We adopt a statistical means of relaxing constraints by introducing random errors with the aim of dithering systematic errors and recovering accuracy in a statistical sense. Given an equality constraint $f(\kappa) = 0$ and its approximation $\tilde{f}(\kappa) = 0$, we derive a relaxation $|\tilde{f}(x)| \leq \epsilon$. For each constraint f some statistics have to be gathered on ϵ so as to minimize the overhead on the size of $\mathcal{K}$ for introducing it.

Once we have this set of constraints, we need to take them into account to define the space of *interesting parameters*. Analog Constraint Graphs (ACGs) are introduced as a bipartite graph representation of configuration constraints. One set of nodes corresponds to equations, the other to variables κ. Bipartite graphs are a common form for dealing with system of equations [7]. A maximal bipartite matching in the ACG is used to compute an evaluation order for equations that is then translated into executable code capable of generating configurations in $\mathcal{K}$ *by construction*. In our experience, even with reasonably straightforward constraints, ratios of the order of 10^{-6} were observed for $\frac{\text{size}(\hat{\mathcal{K}})}{\text{size}(\mathcal{K})}$ with $\mathcal{K} \subset \mathbb{R}^{16}$.

When we deal with the intersection of achievable performance and performance constraints in the top-down phase of the design, we can add to the set of constraints we use to restrict the sampling space the performance constraints so that the results reported above are even more impressive.

Platform	dim($\mathcal{I}$)	dim($\mathcal{O}$)	#Sim	Time
Telescopic	16	6	2134	9h
Folded Cascode	17	6	2838	12h

Table 1: Characterization process results — Performance models are generated on a Sunblade1000.

4. CASE STUDY

We selected a high performance 14 bit, 80 MS/s ADC in 0.13μm CMOS process by STMicroelectronics. Our goal is to investigate the possible tradeoffs between first stage residue amplifier accuracy (gain and linearity) and digital calibration strategies for non-idealities compensation to minimize power consumption. To do so, we hierarchically built a suitable platform library for the ADC, and perform optimizations across the analog/digital boundary.

First stage residue amplifiers are the most power-hungry blocks in high performance pipeline ADCs. The accuracy required in terms of gain and linearity are maximum and calls for large biasing currents in the analog domain. Many techniques are available to estimate and compensate non idealities at the expense of extra digital complexity and power consumption. In this case study we characterize analog components based on multiple amplifier topologies and a digital correction component based on multiple inversion algorithms to find an optimal tradeoff between performances and power consumption. The platform hierarchy is shown in Fig. 2.

4.1 Analog Platform

Two operational transconductance amplifier (OTA) topologies were characterized as illustrated in the bottom-left corner of Fig. 2. At this level of abstraction, the platform provides a Continuous Time (CT) model and a performance model $\mathcal{P}_{OTA} = \mathcal{P}_{fold} \cup \mathcal{P}_{tele}$. Performance models were obtained according to the characterization process described in this paper. Fig. 1 reports performance spaces projection for both topologies. Feasible configuration spaces have been defined for both topologies, leading to less than 3,000 simulations to generate performance models. The ACG used for the telescopic OTA is shown at the top left of Fig. 2. It corresponds to 18 equations and 109 inequalities in 32 variables. The characterization process results are reported in Table 1 and performances are summarized in Table 2. Since OTA interfaces are fixed in this design, all instances are composable by construction ($\mathcal{L}$ is always satisfied). At the same level of abstraction (CT domain), switches were characterized bottom-up to provide accurate charge injection models. Multiple switching architectures may be inserted at this level with different clocking schemes and capacitance values, even if in this case study other design decisions were already taken to make it useless.

Composition of CT OTAs and switch generates a higher platform at the Discrete Time level, which provides a suitable model for the switched capacitor implementation of the interstage amplifier. The new platform exports an architecture space that is the combination of OTA and switch platforms.

4.2 Digital Platform

The digital platform consists of two components, an estimation block and a polynomial inversion block. Together, they implement the digital calibration backend. The estimation block estimates analog non linearity based on a third order polynomial model and the solution introduced in [8]. The estimation algorithm was characterized and its accuracy exported to the estimation platform as well as power consumption. Four inversion algorithms were devised to invert the polynomial non-linearity. They constitute different accuracy versions of an inversion scheme based on a predictor/corrector approach. Both power consumption and accuracy estimates are provided (with bottom up extraction from synthesized circuits) as part of the inversion platforms.

Parameter	Telescopic	Folded
Gain (A_{v0})	$26 \rightarrow 920$	$490 \rightarrow 3100$
Bandwidth (BW) (MHz)	$1.5 \rightarrow 8.9$	$0.29 \rightarrow 2.7$
RMS noise (V_{no}) (mV)	$1.4 \rightarrow 6.2$	$2.8 \rightarrow 29$
Power (mW)	$38 \rightarrow 170$	$56 \rightarrow 180$
Slew rate (SR) (V/μs)	$400 \rightarrow 2,400$	$840 \rightarrow 3,800$

Table 2: Feasible performance ranges of generated configurations

Performance	Optimal	NN1	NN2	NN3
DNL (LSB)	0.4	0.6	0.69	0.24
INL (LSB)	0.1	0.21	0.25	0.58
SNR (dB)	86.3	86.4	86.3	85.8
P_{ADC} (mW)	52.5	52.6	55.8	59.2
P_{OTA} (mW)	47.7	47.9	51.6	55
A_{v0}	220	214	203	184
BW (KHz)	3,380	3,300	3,350	3,290
V_{no} (mV)	2.5	2.61	2.61	2.63
SR (V/μs)	900	903	965	896
G	7.3	7.36	7.31	7.14
P_{Dig} (mW)	4.8	4.8	4.2	4.2

Table 3: Performances of optimal ADC, OTA (telescopic architecture) and digital platform instances as well as their nearest neighbors (NN). G is the amplifier closed loop gain. NN are not far from optimization results, which shows that performance extrapolation in $\mathcal{P}$ was accurate.

4.3 System

The system is finally obtained by composing the digital and analog platforms. It provides a performance space that is the composition of all lower level platforms, thus including both analog and digital alternatives. Model composition is a seamless process since both platforms provide behavioral models at the DT level.

4.4 Optimization and Results

The final design space exploration is formulated as the following optimization problem:

$$\min_{y_{OTA}, y_{GDEC}} \{\theta_1 \cdot P_T + \theta_2 \cdot N_i + \Omega_1(\text{DNL}-0.3) + \Omega_2(\text{INL}-0.7)\}$$

$$\text{s.t.} \quad \begin{cases} \mathcal{P}_{OTA}(y_{OTA})=1, \ \mathcal{P}_{GDEC}(y_{GDEC})=1 \\ DNL<0.7\,\text{LSB}, \ INL<0.9\,\text{LSB} \\ SNR>74\,\text{dB}, \ \epsilon_{corr}<0.5\text{LSB} \end{cases}$$

where the parameters θ are chosen to emphasize minimization of total power consumption P_T. N_i is the total input noise power used to compute SNR for a full scale input signal and ϵ_{corr} is the error due to polynomial inversion. Performances are evaluated through the platform model $\mathcal{F}$. The $\mathcal{P}$ constraints in the formulation enforce architectural constraints; the remaining equations constrain minimum per-

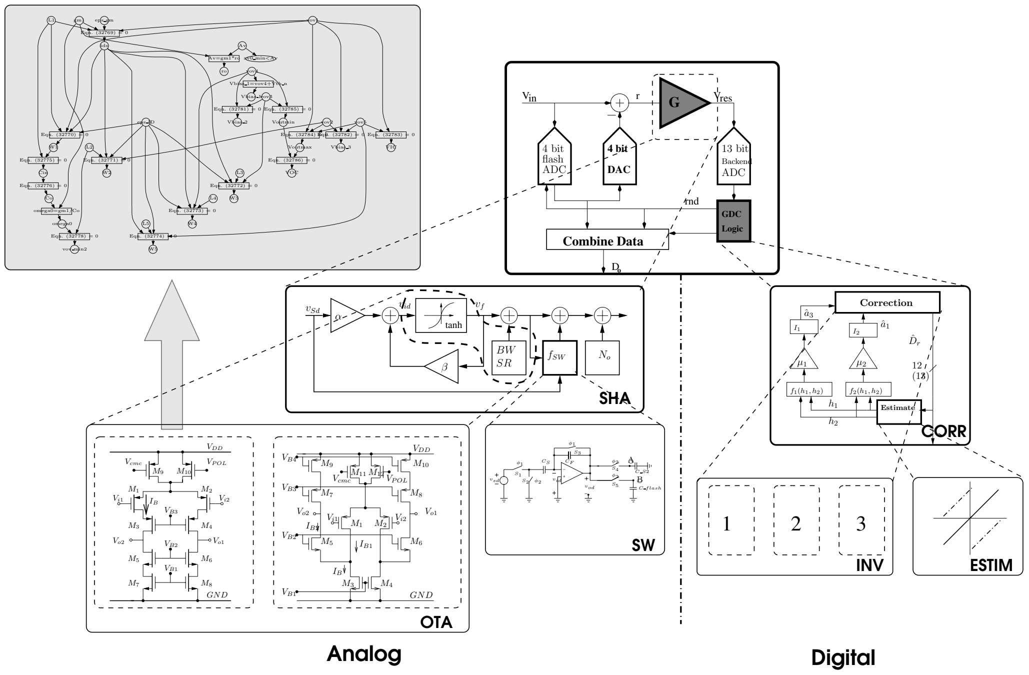

Figure 2: Platform stack for the ADC—Performance annotated blocks, the residue amplifier and the gain and distortion error correction logic (GDEC) are in grey. In the top-left corner the ACG for the telescopic topology is reported.

formance requirements for the converter based on system level requirements.

Optimization was efficiently performed with simulated annealing at the behavioral level; results are reported in Table 3. Design space exploration selected an optimal tradeoff exploiting the telescopic architecture together with a moderately accurate implementation of the inversion algorithm. Power consumption was reduced by 67% at the OTA level compared to the original design (folded topology). The telescopic topology proved to be better performing than the folded cascode one, and requires a simple digital correction scheme. Estimation and correction logic introduce a small contribution to overall power consumption so that power reduction is 64%, demonstrating the superiority of mixed signal approaches to challenging analog designs. Optimal amplifier performances are shown in Table 3 together with the table look-up results for $\mu(y_{opt})$ (Nearest Neighbors OTA configurations). Even though the lower gain of the optimal telescopic OTA introduces larger static errors (with respect to the folded topology), the gain bandwidth product is larger and allows closed loop performances compatible with digital correction. Overall, systematic codesign of analog and digital platforms allowed trimming power consumption to levels comparable to the open loop solution proposed in [8], while still exploiting the advantages of a closed-loop architecture.

5. CONCLUSIONS

We proposed a platform-based design methodology for mixed signal designs. Platform-based Design (PBD) is a meet-in-the-middle design flow that enables architectural tradeoffs to be explored at the behavioral level. In the bottom-up phase, accurate performance model are exported at the system level where critical design decisions can be taken efficiently. We presented some details on how performance models are approximated using Support Vector Machines. The process is completely automated and optimized by exploiting the knowledge of the component structure to reduce the dimensions of the sampling space used to define SVM approximations. The design process is uniform across the analog/digital divide. Therefore, mixed signal designs can be cast into a unified exploration problem. We applied our methodology to a pipeline ADC, optimizing the first stage residue amplifier and the digital backend concurrently, and obtaining 64% power savings over an industrial manually optimized design.

6. REFERENCES

[1] H. Liu, A. Singhee, R. Rutenbar, and L. R. Carley, "Remembrance of circuits past: Macromodeling by data mining in large analog design spaces," in *Proceedings of DAC*, 2002.

[2] T. Kiely and G. Gielen, "Performance modeling of analog integrated circuits using least-squares support vector machines," in *Proc. of DATE*, 2004.

[3] G. Stehr, H. Graeb, and K. Antreich, "Analog performance space exploration by fourier-motzkion elimination with application to hierarchical sizing," in *Proc. of ICCAD*, 2004.

[4] A. Sangiovanni Vincentelli, L. Carloni, F. De Bernardinis, and M. Sgroi, "Benefits and challenges for platform-based design," in *Proceedings of DAC*, pp. 409–414, June 2004.

[5] F. D. Bernardinis, M. I. Jordan, and A. S. Vincentelli, "Support Vector Machines for Analog Circuit Performance Representation," in *Proceedings of DAC*, June 2003.

[6] J. Platt, "Sequential minimal optimization: A fast algorithm for training support vector machines," Tech. Rep. MSR-TR-98-14, Microsoft Research, 1998.

[7] P. Bunus and P. Fritzson, "A debugging scheme for declarative equation based modeling languages," in *Practical Aspects of Decl. Languages : 4th Int. Symp.*, p. 280, 2002.

[8] B. Murmann and B. E. Boser, "A 12-bit 75-MS/s, Pipelined ADC Using Open-Loop Residue Amplification," *IEEE JSSC*, vol. 38, pp. 2040–2050, December 2003.