A Reuse-based Framework for the Design of Analog and Mixed-Signal ICs

Rafael Castro-López, Francisco V. Fernández, and Ángel Rodríguez Vázquez
Instituto de Microelectrónica de Sevilla, Centro Nacional de Microelectrónica
Edificio CICA. Avda. Reina Mercedes s/n, E-41012- Sevilla, Spain
Tel.: +34 955 056 666. Fax: +34 955 056 686
E-mail: Rafael.Castro@imse.cnm.es

ABSTRACT

Despite the spectacular breakthroughs of the semiconductor industry, the ability to design integrated circuits (ICs) under stringent time-to-market (TTM) requirements is lagging behind integration capacity, so far keeping pace with still valid Moore's Law. The resulting gap is threatening with slowing down such a phenomenal growth. The design community believes that it is only by means of powerful CAD tools and design methodologies –and, possibly, a design paradigm shift– that this design gap can be bridged. In this sense, reuse-based design is seen as a promising solution, and concepts such as *IP Block*, *Virtual Component*, and *Design Reuse* have become commonplace thanks to the significant advances in the digital arena. Unfortunately, the very nature of analog and mixed-signal (AMS) design has hindered a similar level of consensus and development. This paper presents a framework for the reuse-based design of AMS circuits. The framework is founded on three key elements: (1) a CAD-supported hierarchical design flow that facilitates the incorporation of AMS reusable blocks, reduces the overall design time, and expedites the management of increasing AMS design complexity; (2) a complete, clear definition of the AMS reusable block, structured into three separate facets or views: the behavioral, structural, and layout facets, the two first for top-down electrical synthesis and bottom-up verification, the latter used during bottom-up physical synthesis; (3) the design for reusability set of tools, methods, and guidelines that, relying on intensive parameterization as well as on design knowledge capture and encapsulation, allows to produce fully reusable AMS blocks. A case study and a functional silicon prototype demonstrate the validity of the paper's proposals.

Keywords: Analog and Mixed-Signal IP Block, Design Reuse, Design for Reusability, CAD.

1. INTRODUCTION

A serious threat to the amazing evolution of the semiconductor industry is the reduction of the design productivity, illustrated in Fig.1. The number of available transistors is growing faster than the ability to design them meaningfully, originating a worrisome design gap [1] [2]. Stringent TTM demands, a very critical factor for ASICs and SoCs that eventually end up in any market place (if the vendor misses the initial market window, prices, and, therefore, profit can be seriously eroded), the continued "consumerization" of the electronic marketplace, and the increasing complexity of designs themselves (not only the increasing number of transistors per chip, but also the new signal processing algorithms and system architectures, the larger design teams, and the steadily shrinking fabrication processes, altogether make the design process far more complex than only a few years ago), are factors indicating that this gap is far from being an incidental situation. The key to deal with this trade-off, i.e., correctly managing design complexity while meeting TTM goals, lies in adopting appropriate and well-structured design methodologies which must be supported by efficient CAD methodologies and tools [3] [4].

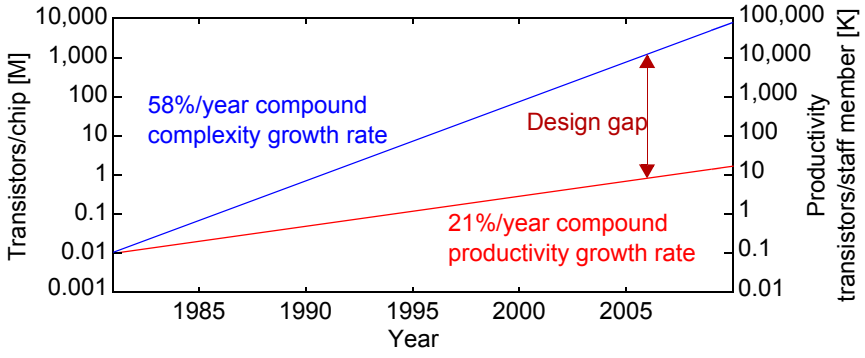


Figure 1: The design gap between complexity and productivity.

VLSI Circuits and Systems II, edited by José Fco. López, Francisco V. Fernández,
José María López-Villegas, José M. de la Rosa, Proceedings of SPIE Vol. 5837
(SPIE, Bellingham, WA, 2005) 0277-786X/05/$15 · doi: 10.1117/12.607930

Although the level of computer digital design automation is far from the push-button stage in the digital arena, the development of digital CAD tools and methodologies are moderately keeping pace with chip density[1]. Unfortunately for AMS CAD resources, the picture so far is much less optimistic, lagging several generations behind the digital ones. A main reason for this detainment is the specialized knowledge and almost handicraft design skills, usually acquired after many years of experience, that analog and mixed-signal design entails, the wide heterogeneity of AMS circuits blocks and designs, the looseness of the hierarchy definition, as well as their extreme sensitivity to several sources of noise such as layout-induced parasitics. Due to all these features, AMS design is still regarded as less systematic, far more knowledge-intensive, and heuristic than digital design. A large amount of analog and mixed-signal circuits are still designed without any really robust and commercial CAD tools but a SPICE-like simulation shell and an interactive and scarcely automated layout environment [3]. Consequently, as fabrication technology progresses and new technology nodes come out, AMS designers are unable to deal with increasing TTM pressures and design complexity, turning AMS design into a clear and present bottleneck of the overall design process.

Considering estimates indicating that 30% of today's ICs include some analog or mixed-signal content (this percentage is expected to grow to over 60% in the next few of years), and the fact that the time required to design AMS content usually dominates the total design time, it is clear that, for the semiconductor industry to maintain its design productivity, an efficient CAD methodology and adequate CAD tools to design AMS circuits are urgently required. In this sense, a number of goals to improve modern top-down, bottom-up, hierarchical design methodologies have been defined in the 2001 *ITRS* report [4]. Among them, this paper addresses the following three goals, whose pivotal issue is the shifting of today's AMS methodologies to the reuse-based design paradigm that, together with adequate synthesis tools, is seen as one of the most promising solutions to face stringent time-to-market pressures and increasing complexity of AMS design:

1. **Reuse**. To extend the notion of library-based approaches, and thereby reduce overall design time, reuse and capturing AMS designers' expertise should be a major concern.
2. **Improve automated synthesis**. AMS circuit design typically requires very specific expert knowledge that should be incorporated into the various automated synthesis processes in order to enhance their efficiency. Hierarchical synthesis with performance specification management and transmission, is mandatory.
3. **Avoid iterations**. Spins between electrical and physical synthesis, accounting for many layout-induced sources of degradation, that eventually lead into product-to-market failure, should be reduced or completely removed.

The contents of the paper are organized as follows. Section 2 provides and overview of the reuse-based design concepts in the AMS arena and a syncretic review of previous approaches. Section 3 presents an integral reuse-based design framework for AMS circuits. A case study is analyzed in Section 4. Conclusions are drawn in Section 5.

2. REUSE IN AMS DESIGN

In information technology, **design reuse** is the inclusion of previously designed components in software and hardware. Likewise, the purpose of design reuse in the IC context is to efficiently use a previous successful design experience in another system and/or a different fabrication process. The degree of reuse is measured by the amount of information and experience that is transferred from the successful first design to the subsequent ones and it counters for design retargeting (when only performance specifications change from one design to the next) and design migration (when there is a change of the fabrication technology involved). Design reuse may be regarded as far from a novel idea. From the earliest days, the first task of an engineer facing a new design problem has been to look over old designs to find a circuit topology or an idea that might be successfully applied to the new problem. Yet, effective design reuse takes much more than just fetching pre-designed components out of a library. Not also successfully designed blocks should be well documented, but all design choices, trade-offs, and heuristics used should be somehow captured, prepared, and packed for ready reuse. As this is not likely to be the usual way of proceeding, **design for reusability** methodologies, tools, and services should be available to foster the creation of **reusable blocks** [6]. These blocks are also known as virtual components (VC) or intellectual property (IP) blocks [7].

The notions of AMS design reuse, AMS design for reusability, and AMS IP have been the subject of heated debates (e.g., [8]). Analog IP, like AMS design automation, is lagging several generations behind existing digital technology. Digital IP is well supported by synthesis tools, there is an extensive repository of available digital IP blocks, and most well-accepted,

1. Nevertheless, a limit is being reached in many areas (e.g., verification) and, consequently, the digital design gap is slowly but steadily widening.

commercial design environments are geared up for these digital IPs. On the other hand, it results naive, if not unrealistic, to expect an exact replica of the digital definitions (i.e., Soft, Firm, and Hard IPs) for the AMS counterpart [2] [5].
The benefits of reuse are, however, large in the long term. It has been estimated that integrating a reusable block into a design means 10 to 100 times less effort than designing the block from scratch [6]. Note that this benefit is only derived as long as the investment to make the block reusable has been made. Some authors think that there is a compelling argument for design reuse: we simply will not be able to build tomorrow's chips without it. Despite the controversy, what most authors agree upon is that AMS IP is useless unless intents and decisions of the previous designer are captured. Thus, full capture and filing of the constraints that experienced AMS designers insist on, is essential for subsequent reuse. Last, it is also worth noting that compliance with widespread-use design environments (e.g., Cadence's Design Framework II™ or Mentor Graphics' Falcon Framework™) should be also pursued in order to minimize a possible hesitancy towards this reuse-based paradigm shift. In summary, the AMS design community still awaits silicon proof that automation and reuse do not come at the expense of harmed performance.

2.1 Recent approaches to analog reuse

Table 1 shows a review of previous approaches to AMS design reuse, some of them aiming at only a few related issues. For each reference, the review has been made attending to the following issues: the kind of reuse addressed (i.e., design retargeting and migration), if hierarchical synthesis has been aimed, if methods and tools for reusability have been developed, if a definition of the AMS reusable block was aimed, if techniques to avoid iterations of the design process were implemented, and the extent of the integration of the tools and techniques into a commercial design environment.

Table 1: Previous approaches to AMS design reuse

Reference	Design reuse			Design for reusability	Definition of AMS reusable block	Techniques to remove sizing-layout spins	Integration into commercial design environments
	Design retargeting	Design Migration	Hierarchical synthesis				
[2]	Yes	Yes	Yes	Limited	Limited	No	No
[5]	Yes	Yes	No	Limited	Limited	Yes	No
[9]	No	Yes	Limited	Limited	No	No	No
[10]	Yes	No	Limited	No	No	No	No
[11]	Yes	Yes	Limited	Limited	Limited	No	No
[12]	No	No	No	Yes	No	No	Limited
[13]	No	No	Limited	Limited	Yes	No	Limited

None of the above reviewed approaches fully completes a framework for promoting reuse-based design of AMS circuits. In this paper, a framework that encompassed all the issues in Table 1 is presented [14].

3. REUSE-BASED DESIGN FRAMEWORK

The ensuing description of a reuse-based design framework is organized in three sections. First, the design reuse flow is presented, emphasizing the hierarchical synthesis approach. Then, the AMS reusable block is described and its relationship with design reuse is analyzed. Finally, design for reusability is explained, outlining the features that should be implemented in any AMS reusable block and the methods and techniques required to do so.

3.1 The design reuse flow

The design reuse flow is illustrated in Fig.2. It is necessary to define and stick to a hierarchical structure of AMS circuits in order to help systematizing the design flow and improve the process of top-down specification transformation (from general to specific) and bottom-up layout synthesis and verification. From bottom to top, the hierarchy levels are:

- **Device** level, composed of active and passive devices, described by means of specific process-dependent models.
- **Cell** level, composed of blocks with basic functionality such as opamps and comparators. These are composed of device-level building blocks.

- **Module** level, defined as stand-alone functions with robust interface that can be clearly distinguished from its environment [5]. In addition to be composed of cell-level blocks as well as separate devices (not belonging to any cell-level block), it will be assumed, for the sake of generality, that the module level can be further divided into one or more module levels.
- **System** or **sub-system** (if there is a digital sub-system). It comprises separate devices, cell, and module-level building blocks.

3.1.1 Adopted synthesis approaches

The synthesis approaches used in the design reuse flow are the following[2]:

- For system, module, and cell sizing, an optimization-based engine has been chosen [15]. Although this engine, called FRIDGE, is able to interact with any kind of performance evaluation approach, simulation is used for the better accuracy (provided that accurate models are used). The implemented approach is a two-step one: in the first one, statistical optimization techniques are applied, while deterministic ones are applied in the second step. The breadth (i.e., the large set of analog problems the tool can deal with) and the ease at adding designers' expertise to the sizing procedures (which can be done by making use of powerful tools like embeddable C++ programs) are also important aspects that have favored this choice. From this point of view, FRIDGE is an optimization-based sizing approach incorporating the appealing features of knowledge-based ones, which turns out essential for reuse of AMS design expertise.
- For layout generation, a template-based approach has been followed. The reasons for this choice are: (1) layout templates are very efficient at handling design expertise which cannot be easily considered by traditional placement and routing algorithms; (2) layout templates ease the placement phase, since the positions of the blocks in the template are stored according to pre-defined relationships embodying constraints from the layout expert (solutions obtained with optimization-driven methods may reveal this same knowledge at the expense of time-consuming algorithmic techniques); (3) layout templates can be straightforwardly ported, because technological independence can be achieved by writing the procedural template generators that are fully independent of the fabrication process parameters; (4) layout templates permit searching for optimal block parameters while a priori revealing the knowledge needed for evaluation of layout parasitics, which turns out essential for minimizing the number of iterations between sizing and layout generation.

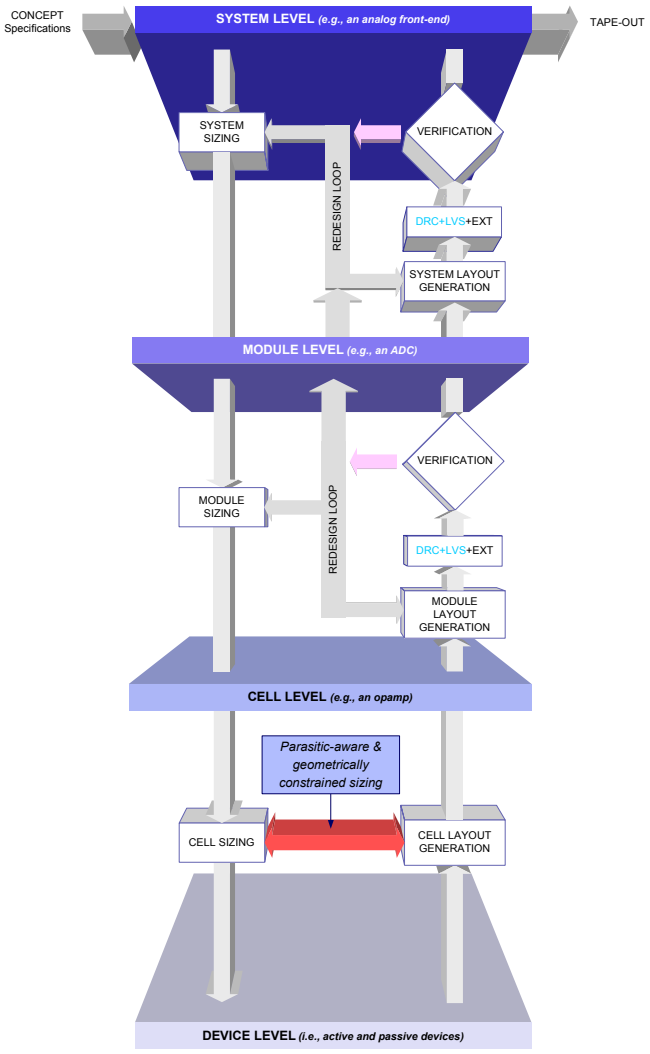


Figure 2: The design reuse flow.

3.1.2 The top-down path

During the top-down path of the design reuse flow, transmission of performance specifications progresses from one level to the levels below. Carried out by using the optimization-based approach described above, three sizing processes take place:

2. As in [2] and [13], it is hereinafter assumed that both the architecture of the system and module-level blocks, as well as cell-level topologies have been decided, so architecture/topology selection is not carried out.

1. **System sizing**, where the system-level specifications are transmitted down to obtain performance specifications for every module and cell-level component of the AMS system, as well as parameters for separate device-level components. If required, and due to the possibly high complexity of the modules and cells at this level, behavioral models are used to speed up the synthesis.
2. **Module sizing**, where the performance specifications of each module-level block are mapped into performance specifications for each of its components (i.e., module and cell-level blocks) and device parameters. As in system sizing, due to the complexity of the circuit at this level, behavioral models are used to speed up the synthesis.
3. **Cell sizing**, where each cell-level circuit is sized to obtain the value of device-level parameters such that the performance specifications for the cell-level circuit are properly addressed.

To reduce iterations between cell electrical and subsequent physical synthesis, consequence of layout-induced parasitics unacceptably degrading the circuit performance, extraction of circuit parasitics is done during the very sizing process. The extraction of parasitics is to be performed by means of either layout generation or parasitic modeling. Heuristic-based or performance-driven approaches are currently too slow for layout generation within the circuit sizing process [16]. Procedural layout template allows, on the other hand, fast generation of circuit layout since no time-consuming optimization algorithms are involved. What is more, templates allow to predict which is going to be the exact shape of the circuit layout without actually moving to the layout phase, therefore allowing parasitic modeling. As built-in circuit devices can be actually implemented in different ways (e.g., resistors and MOS transistors can be folded, the transistors in a differential input pair can be arranged in differing common-centroid arrangements, etc.), this knowledge, previously embedded in the circuit layout template and retrieved at each iteration of cell sizing optimization, can be used to obtain a layout with minimal occupied area and/or featuring a user-specified aspect ratio. Including layout-induced parasitics in the circuit sizing process is known as **parasitic-aware** sizing; including layout geometrical information is called **geometrically-constrained** sizing (since some design variables are constrained for the layout to meet certain user-defined geometric objectives). Note that it is crucial to attack both problems simultaneously, since different device implementation may give rise to several layout-induced effects. The combination of both techniques is called **layout-aware** sizing. The reader is referred to [17] [18] for further details and demonstration examples of this technique.

3.1.3 The bottom-up path

After cell-level sizing, layout is generated by instancing the corresponding template and the obtained sizing solution, which includes the implementation style of each device if geometrical objectives have been defined. No formal verification is required because the layout template is correct-by-construction. Verification of the extracted layout is neither necessary, since layout parasitics have been already considered. In this way, sizing-layout-sizing spins are avoided, thereby speeding up the overall design reuse flow.

Module layout is then generated by assembling the instanced layout of its cell-level components or by instancing the module layout template provided it has been incorporated as a reusable block itself. Only in the former case, formal design rule and layout vs. schematic checks of the module layout are performed. Afterwards, the module performance is verified by using simulation. Provided that the module-level circuit is not very complex, cell-level components can be replaced by their device-level description. Otherwise, each cell-level circuit can be replaced by a corresponding behavioral model, properly backannotated with the attained electrical performance (which includes, as said above, the performance degradation induced by layout). A redesign loop to modify the module layout (i.e., layout elements other than cell layouts) or to repeat the module sizing is initiated in case that the module fails to meet the intended performance specifications.

Bottom-up verification of the system-level circuit follows the same methodology. First, the system layout is generated by assembling the module-level components or by instancing the system-level layout template provided it is available as reusable block. A formal verification precedes the performance verification where behavioral backannotated models of the module-level circuits (including performance degradation due to parasitics) can be used to reduce the simulation time.

3.2 The AMS reusable block

Fig.3 shows the proposed composition of the AMS reusable block. Such conceptual block is composed of three separate, yet linked views or facets, each used at a different design step as design information flows from one view to the other. The three views of the AMS reusable block are:

- **Behavioral** view. This facet comprises a high-level, abstract model of the AMS block as well as the collateral design knowledge required to perform the sizing and verification of the circuit block immediately above in the hierarchy that contains the reusable block.

- **Structural** view. This facet corresponds to a database describing the block in terms of its building components as well as the design knowledge required to perform the sizing and verification of the reusable block itself.
- **Layout** view. The layout facet contains the valuable expertise for the completion of the analog or mixed-signal block's layout.

The facets of the AMS reusable block defined above must be worked out in such a way that smoothly performing both design retargeting and design migration is ensured. This obviously pose several problems that must be solved through a proper design for reusability method. Broadly speaking, whereas the behavioral view is relatively easy to design for reusability, the layout view poses the greatest level of complexity. In a sense, this difference is also present in the digital domain, where hard IPs are very difficult (if not impossible) to reuse for other performance targets or a differing fabrication process; also, the AMS reusable block here should resemble the notion of digital firm IP (an intermediate solution between the very flexible, not predictable soft IP and the very predictable but badly flexible hard IP), retaining its most interesting capabilities, namely flexibility and predictability.

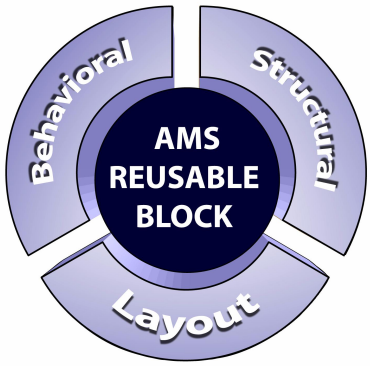


Figure 3: The three facets of the AMS reusable block.

Note the hierarchical nature of the layout and structural facets of the AMS reusable block. While the behavioral facet represents an abstraction of the block performance, the layout facet of a system, sub-system, or module-level reusable block includes the layout facet of their building reusable blocks. For instance, the layout facet of a module-level reusable block comprises the layout facet of any cell-level and module-level reusable blocks below in the hierarchy[3]; in like manner, a sub-system-level reusable block contains the layout facet of the building modules and cells. Regarding the structural facet, any (system, sub-system, or module-level) reusable block incorporates the behavioral facet of its building reusable components. For instance, the structural facet of sub-system-level reusable block incorporates the behavioral facet of its module and cell-level building blocks.

3.3 The design for reusability methodology

The large time/effort to prepare the design entry to both optimization-based electrical synthesis and the knowledge-based layout synthesis approaches used here, is a negative aspect with regard to design reuse[4]. This means that the AMS reusable block must be developed such that design retargeting and design migration are feasible: a third-party user may be able to perform both reuse operations with relatively little effort and faster than without reusable blocks. The AMS reusable block can speed up the synthesis process by providing the necessary input to the synthesis tools. This relevant feature can be achieved thanks to two essential, complementary principles, namely **intensive parameterization** and **design knowledge encapsulation**[5].

Parameterization is the process by which the entire block description is turned completely independent of both the fabrication process and the specific value of the block performance features. That is, by means of parameterization, the AMS block is progressively "released" from a specific performance and from an explicit set of process rules and constraints, becoming, thus, more and more independent. Changing the value of the parameters provides a new performance and expedites the design migration to a different fabrication process. In parameterizing a block there are, however, one obvious limitation: the achievable block performance after parameterization is limited to the achievable performance of the circuit architecture/topology. **Design knowledge** (DK) encapsulation is the process that identifies the relevant design expertise for the block under development and embeds it into the appropriate facet of the AMS reusable block. The design knowledge is fundamental to guide the synthesis tools throughout the design reuse flow.

---

3. In addition, of course, of the remaining, required elements (such as device-level components, routing wires, shielding guard-rings, etc.) completing the layout of the reusable block.
4. Actually, this drawback is common to all synthesis approaches reported so far [14].
5. Finally, another important factor that has been often utterly ignored or underestimated is a thorough documentation of the AMS reusable block. Unless this information is captured, future reuse of the AMS block is limited because the design team faces serious time issues as they struggle to either recreate the information or suffer the problems induced without the benefit of having it available. Each facet of the AMS reusable block must be accompanied with collateral deliverables containing a detailed description of the parameters implemented and the design knowledge encapsulated.

These two principles are applied to each facet, eventually generating a database that the design reuse flow calls for where appropriate. The most significant characteristics of each facet's database are:

- Behavioral facet: the behavioral model of the AMS block must have a design space dimensionality, i.e., a behavioral model deliberately parameterized to cover a range of performance characteristics of one or more circuits (obviously with the same equivalent electrical behavior, such as comparator topologies). These characteristics become the input parameters of the reusable behavioral model. Other properties that the behavioral model must feature are modularity (different performance features can be selectively enabled/disabled depending upon the retargeting experiment and the specifications that are transmitted from the higher level to the reusable block itself) and technological independence (any reference to process data must be generic). DK is also used to select the modeling style (e.g., macromodeling, high-level languages, mathematical models, etc.) and the set of behavioral performance characteristics. Also, a variation range for each performance characteristic must be stored in the database for easy re-definition of the explorable design space.
- Structural facet: this database comprises a parameterized netlist/schematic[6] of the reusable block as well as a design knowledge database used for top-down transmission of specifications from the reusable block to its components and bottom-up verification of the reusable block's achieved behavior. The design knowledge database encompasses the following elements:
    - Design variables, i.e., the independent variables, selected from all parameters of the netlist/schematic, that define the explorable design space for synthesis.
    - Constraints, i.e., relationships that link the rest of the parameters of the netlist/schematic with the design variables. DK is vital in developing these constraints. The sizing engine used in the reuse flow (Section 3.1.2) allows the smart incorporation of constraints that range from simple analytical equations to elaborate design plans.
    - Process data.
    - Testbenches, which are the collections of elements that allow the evaluation of the reusable block's performance, either during electrical synthesis or upon verification. Each testbench contains a set of performance features and monitors (post-processing of the simulation data required in order to extract specific value(s) of the reusable block's performance), peripheral setup (such as loads, supply voltages, bias sources, clocks, and so on), models for the reusable block components, and additional design variables and constraints.
- Layout facet: this database, being the most difficult to complete due to the dependence of the physical representation with technology and performance, relies, as said above, on procedural layout templates. The required features of reusable templates are: parameterized nature, technology independence, relative placement and routing, and hierarchical composition. Furthermore, these features have to comply with requirements that AMS layout experts typically insist on (structures and techniques that improve matching, adjusting wire width and the number of contact/vias to improve circuit's reliability, and so on). Successfully implementing these features can be accomplishing by carefully planning the relative placement and routing through constraint graphs [19] and using a set of geometric and database procedures on the different components of the reusable block's layout [18] [20]. For the present approach, layout templates have been implemented in Cadence's *DFWII* environment, using the capabilities of the PCELL technology [21]. SKILL™ programming [22] has also been used to facilitate the generation of templates, extending the capabilities of PCELL technology to higher levels of the AMS hierarchy [14].

4. CASE STUDY

The IQ DA transmit interface system used in wireless communications [23], whose functional block diagram is shown in Fig.4, is the AMS system used as demonstration vehicle of the reuse-based framework described in this paper. This system provides two fully-differential channels in quadrature phase (90° phase shift) between the input digital ports and the output analog ports. Each channel is formed by a $SINC^3$ digital processing unit for signal shaping and interpolation, a DAC employing current-steering circuit techniques, a continuous-time second-order low-pass filter (CT-LP Filter), and a first-order programmable-gain amplifier (PGA). An additional calibration unit is employed to adjust the unavoidable offsets and

6. Migration of the structural facet's parameterized netlist/schematic from one technology database to another may pose a considerable time delay. For instance, translation of a schematic within the Cadence's *DFWII* environment from one process *kit* to another may become extremely laborious due to the lack of standards between foundries for device-level descriptions. In [14], a re-referencing method is presented that palliates this difficulty.

mismatches between both channels. In order to control the functionality and calibration of the complete system, a control unit is also included.

In the following, the case study is focused on the analog portion of the IQ DA, the analog back-end (the blocks under the shaded area, at the end of the chain, of the IQ DA in Fig.4) composed of the CT-LP filter and the PGA blocks. The purpose of the CT-LP filter, is to attenuate the image components of the baseband spectrum at multiples of the clock frequency, to smooth the output signals generated by the preceding segmented current-steering DAC, as well as to provide current-to-voltage conversion. The purpose of the PGA, on the other hand, is to provide a digitally-controlled DC amplification of the differential output signal delivered by the CT-LP filter, to enforce the image rejection function of the CT-LP filter, and to support the buffering of its output differential signal to the external load of the chip, in order to directly drive commercially available RF modulators with no need of external components.

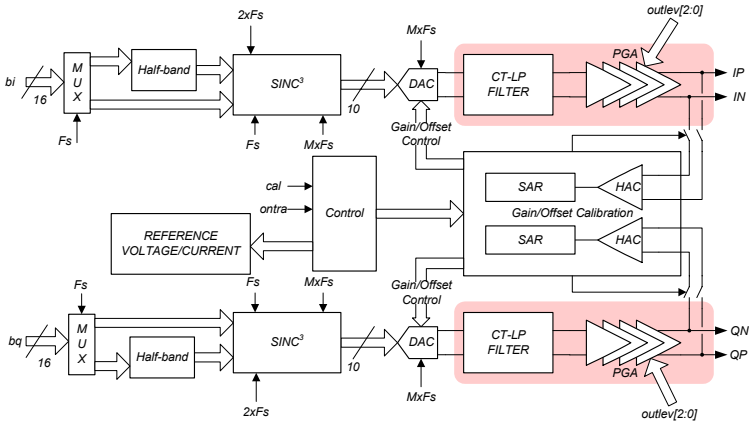


Figure 4: Functional diagram of the IQ DA transmit interface.

4.1 Specifications of the analog back-end

Table 2 shows the performance specifications of the analog back-end for the GSM wireless communication standard.

It is important to note that some of the specifications of the analog back-end involve the concept of design centering. It implies that the analog back-end has to be carefully designed not only to be optimal with respect to nominal performance, but also with respect to manufacturing process variations. These variations can be local (also known as **intra-die** variations), meaning that they affect every device in the die in a different way, and global (also known as **inter-die** variations), which affect every device in the die exactly in the same way. Eventually, their influence results in a mismatch or difference in how channels I and Q process the signal.

Table 2: Performance specifications of the analog back-end.

Feature	GSM standard spec.
Input bit rate	1083.3 kbit/s
Oversampling factor, M	12
Sampling frequency, FS	13 MHz
Signal bandwidth, SB	100 kHz
Minimum attenuation of the analog back-end @FS	49.72 dB
Maximum I/Q gain mismatch	± 0.7 dB
Maximum group delay deviation (from DC to SB)	8.3 ns
Maximum I/Q group delay mismatch (from DC to SB)	33.5 ns
Maximum I/Q phase mismatch @SB	1.3 º

4.2 Hierarchy of the analog back-end

Fig.5 illustrates the hierarchical organization of the analog back-end. As said above, the analog back-end (the sub-system level) is composed of two filtering blocks, a CT-LP filter and a PGA (both at equivalent module levels). These blocks are composed of passive devices (for the feedback and feedforward paths) and amplifying blocks A_1 and A_2 (at the CT-LP filter cell level) and A_3 (at the PGA cell level)[7].

Each block of the analog back-end was made reusable by applying the design for reusability outlined above. Complete synthesis of the analog back-end involves, first, the mapping of the sub-system performance specifications in Table 2 into specifications for each of the module-level components, i.e., the CT-LP filter and the PGA. For this, the structural facet of the analog back-end together with the behavioral facets of each module are used. The CT-LP filter is modeled as an ideal second-order Butterworth filter, whose input parameters are the DC gain, $Kctf$, the pole frequency, $fpolectf$, and the quality factor, Q. In a similar way, the PGA is modeled as an ideal first-order filter with gain, $Kpga$, and pole frequency, $fpolepga$, as input parameters. To include statistical variations in order to account for the analog back-end performance specifications, these parameters are to be considered as the mean value of corresponding Gaussian distributions, therefore

7. Biasing circuitry not shown.

adding $Kctf\_del^{intra}$, $fpolectf\_del^{intra}$, Q_del^{intra}, $Kpga\_del^{intra}$, and $fpolepga\_del^{intra}$ as model parameters modeling the standard deviation characteristic, for the intra-die statistical scenario, and $Kctf\_del^{inter}$, $fpolectf\_del^{inter}$, Q_del^{inter}, $Kpga\_del^{inter}$, and $fpolepga\_del^{inter}$ for the inter-die statistical scenario. With the exception of $Kctf$, Q, and $Kpga$, externally fixed to V_{FS}/I_{FS}, (with V_{FS} and I_{FS} being the full-scale voltage and current respectively), $1/\sqrt{2}$ (for a maximally-flat pass-band), and $\{1, 2/3, 1/3\}$ (the required PGA DC amplification levels controlled by the digital word outlev[2:0]), the objective of the analog back-end sizing is the maximization of the standard deviation values; that is, the goal is to look for the values of $fpolectf$ and $fpolepga$, as well maximum values of $Kctf\_del^{intra}$, $fpolectf\_del^{intra}$, Q_del^{intra}, $Kctf\_del^{inter}$, $fpolectf\_del^{inter}$, and Q_del^{inter} such that the analog back-end is centered with respect to its performance specifications.

Second, these module-level performance specifications are to be translated into values of the passive elements, on the one hand, and performance specifications for each of the three opamps, on the other hand. Both processes must be carried out taking into account the inter- and intra-die two scenarios. For the CT-LP filter, the design variables are the passive components characteristics and the performance specifications of opamps A_1 and A_2 (see Table 4 below); the goal is then to find adequate values of these variables such that the previously obtained specifications in $fpolectf$, $Kctf\_del^{intra}$, $fpolectf\_del^{intra}$, Q_del^{intra}, $Kctf\_del^{inter}$, $fpolectf\_del^{inter}$, and Q_del^{inter} are fulfilled, which also implies to consider both the intra-die and the inter-die scenarios. Sizing the PGA follows a similar methodology.

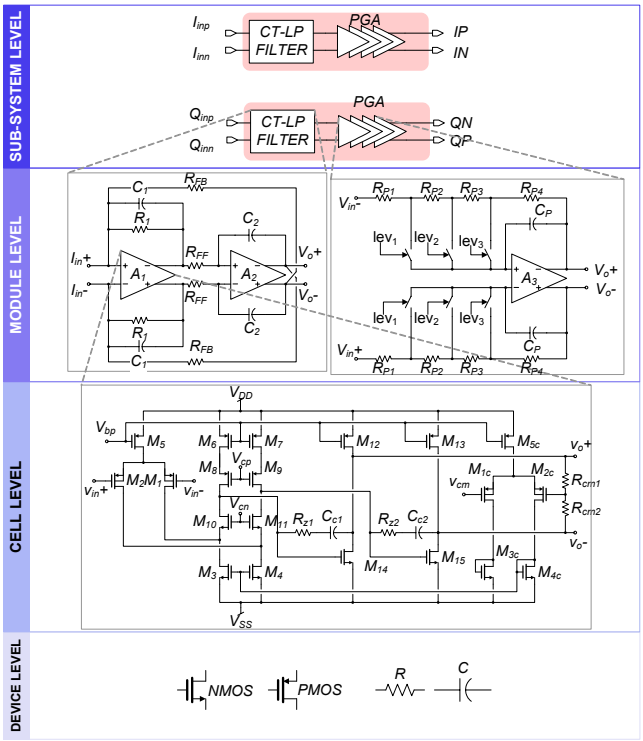


Figure 5: Hierarchical levels of the analog back-end.

For the different sizing tasks, the optimizer FRIDGE [15] has been used in combination with HSPICE™ simulator [24]. For layout generation, the Cadence® technology has been employed (i.e., the SKILL™ programming language and the Virtuoso® layout editor [25]). In like manner, formal layout verification (i.e., design-rule checking, extraction, and layout-*vs*.-schematic tools) is done within the CADENCE's *Design FrameWork II* environment. Verification of the synthesized design is also done within such framework.

4.3 Reuse of the analog back-end in a 0.35-μm CMOS process

Turned fully reusable, the analog back-end can be rapidly retargeted and migrated to different wireless standard communication specifications and to any sub-micron CMOS technology with 2.5 V/3 V/5 V supply and a minimum of three metal and two poly layers. The reuse process reported here takes place for GSM (whose specifications are listed in Table 2) and in a 0.35-μm CMOS technology. The final results of the reuse experiment are collected in Table 3. The third column in this table collects the results from the system-sizing process, with which module-sizing is undertaken. Final verification results, including all layout-induced parasitics, are shown in fourth column of Table 3. This verification was performed at full device level due to the relative low complexity of the modules and cells, the required simulations, and the accuracy requirement. Table 4 and Table 5 show intermediate results of the reuse process. Table 4 show the required resulting passive devices, which were obtained after each module sizing process. Table 5 shows design information regarding the opamps A_1, A_2, and A_3. The 'Specified' columns refers to performance values that were obtained after module-sizing.

Table 3: Final analog back-end verification for the GSM standard.

Performance feature	Specification	Optimized during system sizing	Value	Units
Maximum I/Q gain mismatch	< 0.7	0.48	0.011	dB
Maximum group delay deviation (from DC to SB)	< 33.5e-9	8.45e-9	1.43e-9	s
Maximum I/Q phase mismatch @SB	< 1.3	0.3	0.05	º
Minimum attenuation of the analog back-end @FS	> 50.0	57.37	66.64	dB
Maximum I/Q group delay mismatch (from DC to SB)	< 8.3e-9	8.29e-9	7.54e-9	s

The 'Obtained' column refer to performance values including the degradations induced by layout parasitics, since layout-aware synthesis were applied.

Table 4: Required passive devices.

Device	Value	Units
R_1	41.355	kΩ
R_{FF}	798.110	kΩ
R_{FB}	4.286	kΩ
C_1	3.728	pF
C_2	3.728	pF
R_{P1}	25.0	kΩ
R_{P2}	5.0	kΩ
R_{P3}	7.5	kΩ
R_{P4}	12.5	kΩ
C_P	3.984	pF
C_P	0.995	pF

The resulting layout instance of the analog back-end for this GSM standard retargeting is depicted in Fig.6. Total occupied area is $750.4 \times 1250.6\ \mu m^2$. The last step of the design reuse flow is verification, which, as said above, was directly performed using full device-level simulation with extracted parasitics[8]. Performed on a PentiumIV@1.3GHz PC with 512Mb RAM, total retargeting time was 62.2 minutes CPU time, which includes 15 minutes for analog back-end and module sizing, 14.6 minutes for cell-level sizing, layout generation, extraction, and verification, and about 31 minutes for layout generation (10s), extraction (30s), and verification (30 minutes) of the analog back-end.

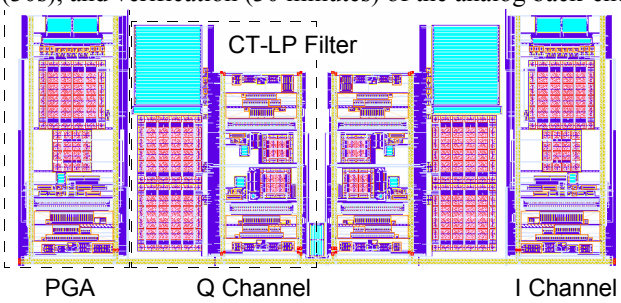


Figure 6: Analog back-end final layout instance.

A prototype of the complete IQ DA transmit interface, whose analog back-end has been completed by following the design reuse methodology presented in this paper, was fabricated in a double-poly, 3.3V supply voltage 0.35-µm CMOS technology. Fig.7 shows a microphotograph of such prototype. It occupies (pads excluded) $2.8mm^2$. The experimental results are summarized in Table 6, which reveal that the system operates correctly.

5. CONCLUSIONS

A framework for the reuse-based design of AMS circuits has been described. This framework includes a clear definition of the AMS reusable block's characteristics, for which a design for reusability methodology has been outlined. Relying on optimization-based, simulation-in-the-loop electrical synthesis and procedural physical synthesis via templates, the design reuse flow has been illustrated with an analog case study, the back-end of an industrial-scale IQ DA transmit interface, a prototype of which was subsequently fabricated in a 0.35-µm CMOS process.

The development cost (necessary effort to develop the reusable block with the design for reusability methodology) of design for reusability of the analog back-end has been evaluated in nearly 6 persons · month , from which around 50% is

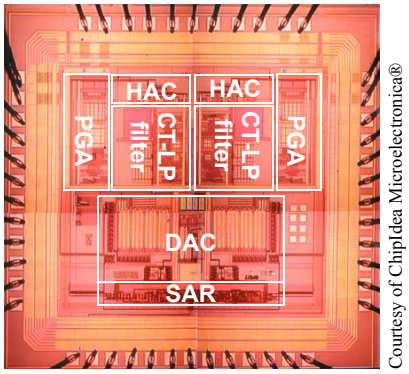
8. Though, strictly speaking, verification of modules was should take place after cell verification, the nature of the analog back-end and the flexibility of the design reuse flow itself allow to skip module verification.

Table 5: Obtained opamp performances after cell-level sizing.

Feature	Opamp A_1		Opamp A_2		Opamp A_3		Units
	Specified	Obtained	Specified	Obtained	Specified	Obtained	
DC Gain	> 50	101.7	> 60	99.5	> 60	93.87	dB
Unity-gain frequency (ft)	> 26.43	29.26	> 43.92	43.93	> 51.41	52.36	MHz
Phase margin	> 72.83	83.01	> 39.05	86.21	> 47.78	57.03	º
Output resistance	< 40	36.26	< 70	67.94	< 70	69.98	kΩ
DC Gain of the CMFB	> 50	111.4	> 60	106.8	> 60	107.5	dB
Unity-gain frequency of the CMFB	≈ ft	28.86	≈ ft	42.86	≈ ft	52.96	MHz
Phase margin of the CMFB	> 72.83	74.67	> 39.05	74.23	> 47.78	57.77	º
Load resistance	36.5	--	6.0	--	0.97	--	kΩ
Load capacitance	3.9	--	3.8	--	100.35	--	pF
Power	Minimize	1.07	Minimize	0.72	Minimize	0.86	mW
Area	Minimize	977.337	Minimize	984.39	Minimize	1648.67	μm^2

Table 6: Experimental results summary.

Characteristic	Specified	Measured			Units
		Min.	Average	Max.	
SINAD	57	60.91	65.42	71.28	dB
SNR	60	63.29	70.50	75.28	dB
THD	-60	-62.23	-67.44	-73.89	dB
SFDR	65	66.29	70.84	97.72	dB
DNL	±1.0	-1.29	-0.21	0.87	LSB
INL	±1.0	-1.42	-0.28	0.86	LSB



Courtesy of ChipIdea Microelectronica®

Figure 7: Microphotograph of the prototype in a 0.35-µm CMOS technology.

devoted to developing the reusable opamps and roughly 50% of the cost for each reusable block was devoted to the development of the its layout facet. Prior to the design for reusability of the analog back-end, it took about 4 persons · month to complete its design following a traditional design flow. The investment that the extra development cost on design for reusability entails is, on the other hand, to be recovered in the lifetime reuses of the design. The running times for the complete reuse of the analog back-end circuit (once it is completely reusable) takes, from sizing to final verification, around 60 minutes of CPU time. Both, the case study and the silicon prototype, thus confirm that the application of the reuse-based design framework to a AMS circuit will certainly dilute the initial investment, rendering the difference in efforts increasingly irrelevant.

ACKNOWLEDGMENTS

This work has been partially supported by project TEC2004-01752, funded by the Spanish Ministry of Education and Science.

REFERENCES

1. *Int. Technology Roadmap for Semiconductors. Edition 1999*. Available: http://public.itrs.net/files/1999_SIA_Roadmap/Design.pdf, 1999.

2. J. Vandenbussche, G. Gielen, and M. Steyaert, *Systematic design of analog IP blocks*. Boston: Kluwer Academic Publisher, 2003.
3. G. Gielen and R. A. Rutenbar, "Computer-aided design of analog and mixed-signal integrated circuits," *Proc. IEEE*, vol. 88, no. 12, pp. 1825-1854, December 2000.
4. *Int. Technology Roadmap for Semiconductors*. *Edition 2001*. Available: http://public.itrs.net/Files/2001ITRS/Home.htm, 2001.
5. M. Dessouky, *Design for reuse of analog circuits. Case study: very low-voltage delta-sigma modulator*. Ph.D. Thesis, University of Paris VI, 2001.
6. M. Keating and P. Bricaud, *Reuse methodology manual*, 2nd ed. Boston: Kluwer Academic Publishers, 1999.
7. *Virtual components design reuse,* R. Seepold and N. Martínez Eds. Boston: Kluwer Academic Publishers, 2001.
8. R. J. Koch and F. Dielacher, "Analog IP - stairway to SoC heaven?," in *Proc. of IEEE Int. Solid-State Circuits Conf.*, 2003, pp. 1-2.
9. K. Francken and G. Gielen, "Methodology for analog technology porting including performance tuning," in *Proc. of IEEE Int. Symp. on Circuits and Systems*, 1999, vol. 1, pp. 415-418.
10. R. Phelps, M. J. Krasnicki, R. A. Rutenbar, L. R. Carley, and J. R. Hellums, "A case study of synthesis for industrial-scale analog IP: redesign of the equalizer/filter frontend for an ADSL CODEC," in *Proc. of ACM/IEEE Design Automation Conf.*, 2000, pp. 1-6.
11. G. Van der Plas, G. Debyser, F. Leyn, K. Lampaert, J. Vandenbussche, G. Gielen, W. Sansen, P. Veselinovic, and D. Leenarts, "AMGIE-A synthesis environment for CMOS analog integrated circuits," *IEEE Trans. Computer-Aided Design*, vol. 20, no. 9, pp. 1037-1058, September 2001.
12. D. Liu, S. Sidiropoulos, and M. Horowitz, "A framework for designing reusable analog circuits," in *Proc. of ACM/IEEE Int. Conf. on* Computer Aided Design, 2003, pp. 375-380.
13. M. Hamour, R. Saleh, S. Mirabbasi, and A. Ivanov, "Analog IP design flow for SoC applications," in *Proc. of IEEE Int. Symp. on Circuits and Systems*, 2003, pp. 676-679.
14. R. Castro-López, *Reuse-based Methodologies and Tools in the Design of Analog and Mixed-Signal Integrated Circuits*. PhD Thesis, University of Seville, 2005.
15. F. Medeiro, B. Pérez-Verdú, and A. Rodríguez-Vázquez, *Top-down design of high-performance sigma-delta modulators*. Boston: Kluwer Academic Publishers, 1999.
16. K. Lampaert, G. Gielen, and W. Sansen, *Analog layout generation for performance and manufacturability*. Boston: Kluwer Academic Publishers, 1999.
17. R. Castro-López, F. V. Fernández, and A. Rodríguez-Vázquez, "Geometrically-constrained, parasitic-aware synthesis of analog ICs", in *Proc. of VLSI Circuits and Systems II Conf.,* to be published, May 2005.
18. R. Castro-López, F. V. Fernández, F. Medeiro, and A. Rodríguez-Vázquez, "Generation of technology-independent retargetable analog blocks," *Analog Integrated Circuits and Signal Processing*, vol. 33, no. 2, pp. 157-70, Kluwer Academic Publishers, November 2002.
19. J. K. Ousterhout, "Corner stitching: a data-structure technique for VLSI layout tools," *IEEE Trans. on Computer-Aided Design*, vol. 3, pp. 87-100, December 1984.
20. R. Castro-López, F. V. Fernández, and A. Rodríguez-Vázquez, "On the suitability and development of layout templates for analog layout reuse and layout-aware synthesis", in *Proc. of VLSI Circuits and Systems II Conf.,* to be published, May 2005.
21. *Virtuoso® parameterized cell reference. Product version 4.4.6*, Cadence Design Systems Inc., 2000.
22. *SKILL language reference. Product version 06.00*, Cadence Design Systems Inc., 2001.
23. J. Franca, N. Horta, M. Pereira, J. Vital, R. Castro-López, M. Delgado-Restituto, F. Fernández, A. Rodríguez-Vázquez, J. Ramos, and P. Santos, "RAPID-retargetability for reusability of application-driven quadrature D/A interface block design," in *Proc. of IEEE Int. Conf. on Electronics, Circuits, and Systems*, 1999, vol. 3, pp. 1679-1683.
24. *HSPICE™ simulation and analysis user guide*. *Release U-2003.03-PA*. Synopsys, Inc., 2003.
25. *Virtuoso® layout editor user guide. Product version 4.4.6*, Cadence Design Systems Inc., 2000.