RUNE: Platform for automated design of integrated multi-domain systems Application to high-speed CMOS photoreceiver front-ends

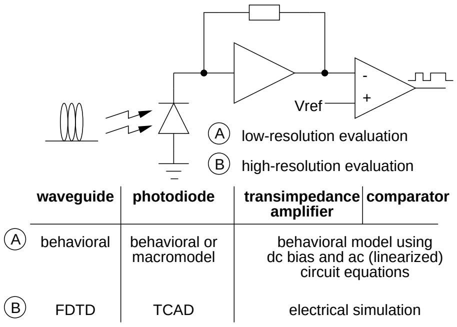
Faress Tissafi-Drissi, Ian O'Connor, Frédéric Gaffiot
Ecole Centrale de Lyon
Laboratory of Electronics, Optoelectronics and Microsystems
36 avenue Guy de Collongue, F-69134 ECULLY cedex, FRANCE
{faress.tissafi-drissi—ian.oconnor—frederic.gaffiot}@ec-lyon.fr

Abstract

In this paper, we present a framework for the automated design of integrated multi-domain systems. The platform allows the designer to set optimization problems according to a hierarchical decomposition strategy, define complex specification functions for each block at a given hierarchical level, follow the progress of optimization and finally view results. Encapsulation of design methodologies is simplified through access to a library of optimization algorithms. The framework is demonstrated through the co-synthesis of a high-speed CMOS photoreceiver front-end comprised of a PIN photodiode and a transimpedance amplifier.

1. Introduction

Evolving system on chip architectures are posing serious design challenges which must be addressed by new methodologies and tools. Two of these challenges are *complexity* and *heterogeneity*. Concerning the complexity, higher integration density and increasing operating frequencies enable the generation of increasingly complex functions and greater processing power. Modern design processes require increasingly abstract levels of definition to manipulate such complex IP blocks. As concerns heterogeneity, integrated systems are progressively taking on board elements of different natures (analog, digital, optical, mechanical ...). Design flows are however segregated (i.e. devices from different domains are designed separately) meaning that the overall system is not optimized and the design process is inefficient. Integrated optical interconnects, and in particular photoreceiver front-ends, are especially representative of relatively new breeds of technology for which existing design technology is inadequate. Fig. 1 shows the receiving end of an integrated optical link. The performance of this link can be simulated (A) with parameterized behavioral component models to verify the functionality at the system level, but this gives no clue as to the physical consequences (area, power, parasitics) of the choice of parameters. Such information can only be obtained by designing the various components and evaluating with methods appropriate to the domain (B). Links to such evaluation methods could in theory be effected through a single high-level simulator [6] implementing multi-domain behavioral description languages such as VHDL-AMS and Verilog-AMS, but in practice this proves difficult and time-consuming.



	waveguide	photodiode	transimpedance amplifier	comparator
A	behavioral	behavioral or macromodel	behavioral model using dc bias and ac (linearized) circuit equations	
B	FDTD	TCAD	electrical simulation	

Figure 1. System schematic

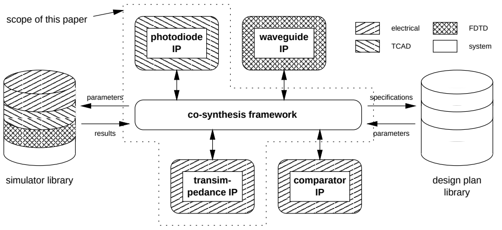


Figure 2. Cosynthesis backplane showing multiple simulators and design plans

Our solution consists of (i) carrying out top-down design space exploration, (ii) physical sizing linking directly from the co-synthesis backplane to the various evaluation tools, as shown in fig. 2, and (iii) subsequent bottom-up design verification using model parameter extraction. Some necessary stages can be attributed to any iterative design cycle. These are:

❖ sizing and iterative adjustment of the parameters of the various sub-blocks of the system to be designed, until the performance criteria at a given level satisfy the specifications,

❖ breakdown of the overall block into sub-blocks and sizing of these sub-blocks,

❖ verification of overall system performance.

In the framework that we have developed, the user can create IP[1] blocks, generic sizing methods, links to evaluation tools and target technology databases. An object-oriented approach is the natural choice for the implementation of this platform due to the ease of adding modules at later stages. We chose the Java language for its portability and also for its dynamic class loading which considerably facilitates on-the-fly equation and procedure development. Section 2 describes the design approach for one hierarchical level. Hierarchy management is detailed in section 3.

2 Single-level design loop

At a given hierarchical level during the synthesis phase, all information concerning the topology under synthesis, design plan and technology are grouped together into one object, which will subsequently be plugged into the sizing/evaluation interfaces. The synthesis flow at one hierarchical level is shown in fig. 3. The *topology* object (IP block) is a key element in the platform. It is comprised of several elements:

❖ synthesis information for specific design methods (an explicit procedure or heuristics, for example).

❖ objective performance indicators, which can be either (i) a system of evaluation equations encapsulated in a behavioral model and formulated in terms of the physical dimensions of the topology, or (ii) a link to a numerical simulation harness (simulation test bench) common to all topologies of one type (*category*), instantiating the topology under certain test conditions and targeting a specific analysis.

❖ individual dimensions: two types are used here, since we make an essential distinction between *abstract* and *physical* dimensions. The former represent the independent design variables that can be extracted from a formal representation of the optimization problem, while the latter are derived (usually explicitly) from the abstract dimensions for evaluation purposes. For example, a CMOS transistor is usually sized (abstract dimensions) by length and W/L ratio to distinguish influences on intrinsic gain and output conductance; whereas for evaluation purposes (physical dimensions) the absolute width and length values are calculated explicitly from the abstract dimension values.

The manner in which the elements in the topology IP are exploited during the design process is formalized by a *design plan*, representing a sequence or a loop of sizing methods. The capability of drawing on a library of homogenized algorithms to build a large range of design plans is attractive, since the user can tailor the plan to the application without having to worry about low-level algorithm code details.

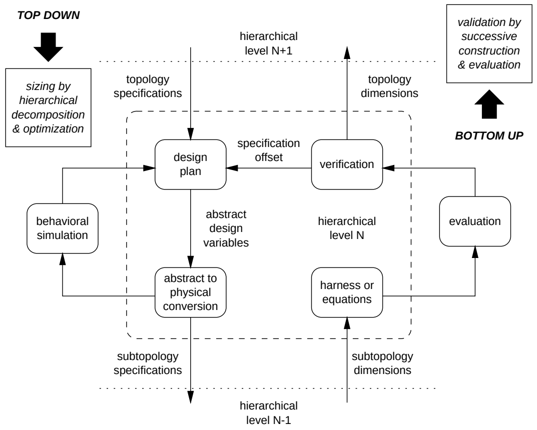


Figure 3. Synthesis flow at one hierarchical level

In general, such plans consist of at least two methods: one to find the zone with the highest probability of containing the global optimum (procedure, mesh, genetic algorithms), and one to accurately and rapidly pinpoint the optimum within the zone (gradient, direct search methods).

Fig. 4 shows what happens between performance evaluation for one set of dimensions, and generation of the following set. The error function is computed from comparison between specified and evaluated performance values, depending also on the type of specification. The current algorithm in the design plan stack is called for a method “hit” (one iteration) based on the algorithm’s tolerances, design history and constraints and (according to user needs) heuristics.

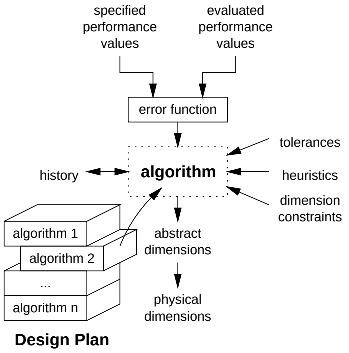


Figure 4. Design plan concept

A new set of abstract dimensions is generated and translated into physical dimensions for evaluation. All sizing method classes inherit from an abstract class [2] encapsulating the “black box” requirements for a method to be able to operate within the platform.

The design objective function itself is built up from a summing of individual performance criteria error functions, of which there can be three types. In the following definitions, ε represents the individual error function contribution

[1] Intellectual Property: we refer here to the encapsulation of any topology specific information that can be used for evaluation or synthesis

[2] Object-oriented terminology here: a class defining method prototypes

of the particular specification, W_i represents the weighting function, P_s the specified performance value and P_r the realized performance value.

✥ constraints (inequalities) which must be satisfied. Their contribution to the error function is evaluated as $\varepsilon_{cs} = W_i|\frac{P_s - P_r}{P_s}|$ while the constraint is unsatisfied, $\varepsilon = 0$ otherwise.

✥ costs to be minimized. Here $\varepsilon_{ct} = \pm W_i \frac{P_s - P_r}{P_s}$ depending on the type of the cost (maximize or minimize).

✥ conditions (equalities) which represent fixed points with tolerances. If the real value is outside the tolerances, then $\varepsilon_{cd} = W_i|\frac{P_s - P_r}{P_s}|$.

The choice of the type of evaluation to be carried out for each individual performance criterion is open to the user. Two types are possible (by equation or by simulation) and can be compared according to three factors: accuracy, CPU time and preparation time. In the platform architecture, the performance class contains a link to execution of a specific analytical equation class, or to running of a particular simulation tool. Each performance object is configured at runtime such that it "knows" how to evaluate itself. For simulation evaluation of a performance criterion, the user creates a simulation harness object which represents the various elements necessary to one simulation: the simulator command, options and analysis type, the harness file, and the post-simulation function to be applied, as shown in fig. 5. Post-simulation functions extract the performance value from the simulation results file. A library of performance evaluation functions has been created, each operating on input and output signals, and some requiring certain accuracy control arguments.

Process independence is guaranteed through the use of a technology class, represented by a file which contains all information concerning process parameters, including device models. The combination of all these elements allows creation of the final netlist for evaluation by simulation. During a synthesis run, the simulator is called on the netlist and generates a results file which must subsequently be converted to standardized tabular form by a simulator-dependent interface. Generation of the simulated performance value is then carried out by simply calling the necessary function from the post-simulation function library.

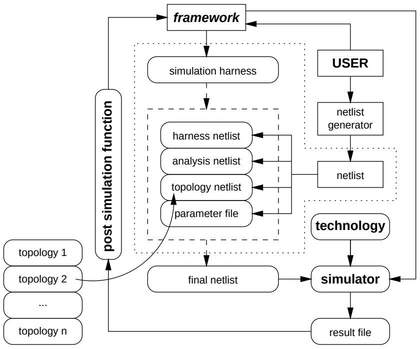
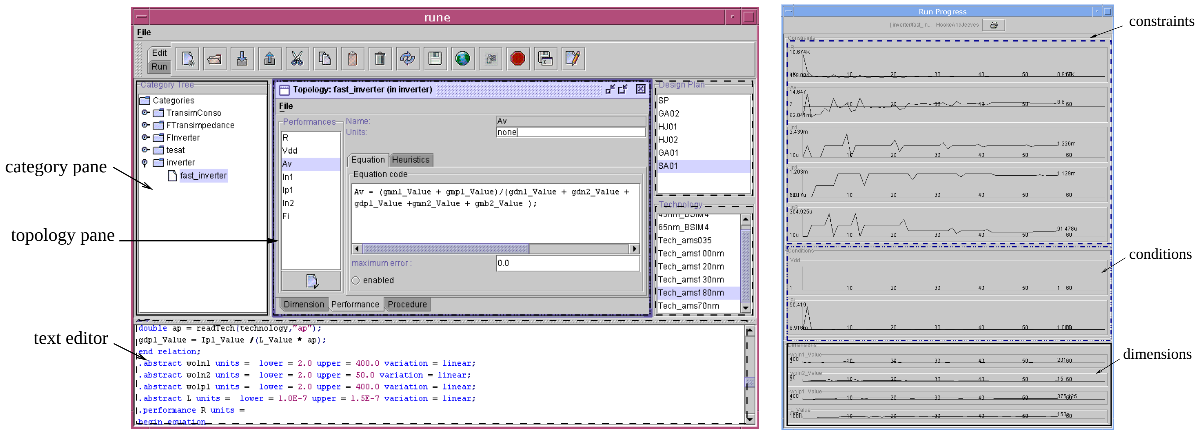


Figure 5. Simulation harness and interface

All functions described in this section have been integrated into the graphical editing tool. Fig. 6 shows the corresponding interface. The interface can be exploited by the designer in two modes of operation: optimize (run the design plan to find a solution in the design space) and evaluate (specify a variable value and evaluate all constraints). With the evaluate mode, the designer can analyse different tolerances in any given design.



Name: Av

Units: none

Equation code

```
Av = (gmn1_Value + gmp1_Value)/(gdn1_Value + gdn2_Value +
gdp1_Value +gmn2_Value + gmb2_Value );
```

maximum error : 0.0

○ enabled

```
double ap = readTech(technology,"ap");
gdp1_Value = Ip1_Value /(L_Value * ap);
end relation;
.abstract woln1 units =  lower = 2.0 upper = 400.0 variation = linear;
.abstract woln2 units =  lower = 2.0 upper = 50.0 variation = linear;
.abstract wolp1 units =  lower = 2.0 upper = 400.0 variation = linear;
.abstract L units =  lower = 1.0E-7 upper = 1.5E-7 variation = linear;
.performance R units =
begin equation
```

(a) Editing Rune interface: Category pane representing the organization of categories and topologies belonging to them; Technology pane representing technologies allowed for design; Topology pane containing all variables, performances and corresponding equation code

(b) Waveform interface: performance evaluation, specification and sizing variables are shown in the run progress form interface

Figure 6. RUNE : fRamework for aUtomated aNy dEsign

3 Illustration of hierarchy management

Our platform processes hierarchically structured systems in a simple and efficient way. The automated synthesis approach proposed by Rune is an automated top-down design flow incorporating bottom-up verification. We will explain this through the design methodology for a high-speed CMOS photoreceiver front-end (fig. 7).

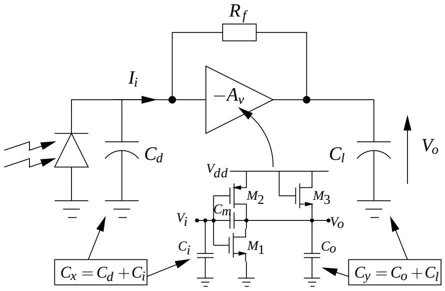


Figure 7. Photoreceiver front-end structure

3.1 Top-Down Design methodology

CMOS photoreceiver front-ends are one of the most critical components in optical links. Such circuits are of profound interest to systems using optical chip-to-chip and on-chip interconnect. Our objective is to implement a design methodology for the design of high-speed CMOS photoreceivers based on a PIN photodiode and transimpedance structure. The PIN photodiode is exposed to a light source of wavelength λ and optical power P_L, and generates a current I_{ph} according to its photoresponsivity R_d. The role of the transimpedance amplifier (TIA) is to convert the photocurrent to a voltage V_o, the whole operating at data rate D. We have used relatively simple blocks in order to demonstrate the feasibility of hierarchical synthesis of the photoreceiver.

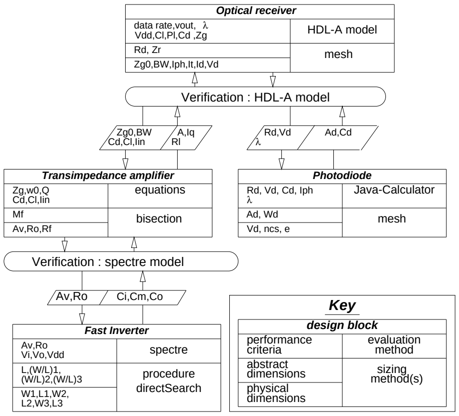


Figure 8. Flow model for receiver synthesis

The top-down design methodology for the photoreceiver is based on two principal ideas. The first is to decompose the photoreceiver system into blocks based on their type and circuit structure complexity. The flow model shown in fig. 8 uses four blocks at three hierarchical levels. The second idea is to define a procedural design methodology for each block, taking into account their respective positions in the hierarchy. The corresponding methodologies are as follows:

➥*Optical receiver*: at this level we represent the receiver with electrical models [1], regardless of the physical structure of the photodiode. For this reason we have used a simple equivalent electrical current source I_{ph} in parallel with the diode capacitance C_d. The transimpedance amplifier TIA is represented by an impedance Z_{in} and a simple linear transfer function:

$$v_{out} = \frac{Z_g}{1 + \frac{s}{2\pi BW_e}} v_{in} \tag{1}$$

where Z_g is the transimpedance gain, and BW_e is the electrical bandwidth. I_{ph} is calculated from $R_d P_L$. One important constraint at this level is the transimpedance load Z_{in}, because a large Z_{in} gives high sensivity and low dynamic range, and the opposite case occurs for small values of Z_{in}. To avoid this conflict, we have chosen abstract dimensions to size the optical receiver with R_d and $Z_r = Z_g/Z_{in}$. The specifications are Z_g and C_d, data rate, dynamic output voltage v_{out}, supply voltage V_{dd}, P_L, λ, and output load capacitance of the photoreceiver. The bandwidth is extrapolated from the data rate D using $D \approx 1.4BW_e$ to retrieve sufficient signal power above the fundamental. The sizing methodologies for the optical receiver are based on a direct search optimization algorithm, for which we require an explicit procedure for starting point generation (fig.9).

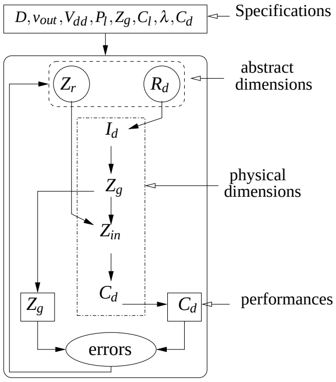


Figure 9. Design procedure for optical receiver

➥*PIN photodiode*: in order to evaluate the photodiode performance during the physical sizing process, we used an internally developed calculator, based on standard PIN photodiode equations from the literature [3]. The specifications are the photoresponsivity R_d, junction capacitance C_d, wavelength λ, reverse bias voltage V_d. We also define material parameters such as the energy gap, absorption coefficient at required wavelength, average carrier mobility, etc. In our case, we have used InGaAs material. The physical dimensions to be used in the sizing process represent the diode structure; intrinsic zone thickness w_d, area A_d.

➥*TIA*: the basic transimpedance amplifier structure in a typical configuration is shown in fig.7. We target CMOS technology and as such we can replace the amplifier block by a model with capacitive input impedance. We model the photodiode simply as a current source with parasitic capacitance [4].The system described is one of second order. The expression for the transimpedance gain Z_g is given by equation 2, where R_o and A_v are, respectively, output resistance and gain of internal amplifier. By introducing the multiplying factors $M_f = R_f/R_o$, $M_i = C_x/C_y$, $M_m = C_m/C_y$ and normalizing all expressions to the time constant $\tau = R_oC_y$, we have expressions for the electrical bandwidth represented by $2\pi\omega_0$(equation 3) and pole quality factor Q (equation 4) [7].

$$Z_g = \frac{R_o - R_f A_v}{1 + A_v} \tag{2}$$

$$\begin{aligned} \omega_0 &= \sqrt{\frac{1+A_v}{R_oR_f(C_xC_y + C_m(C_x + C_y))}} \\ &= \frac{1}{R_oC_y}\sqrt{\frac{1+A_v}{M_f(M_x + M_m + M_xM_m)}} \end{aligned} \tag{3}$$

$$\begin{aligned} Q &= \frac{\sqrt{(1+A_v)(R_fR_o(C_xC_y + C_m(C_x + C_y)))}}{C_x(R_o + R_f) + C_yR_o + C_mR_f(1+A_v)} \\ &= \frac{\sqrt{M_f(M_x + M_m(1+M_x))(1+A_v)}}{1 + M_x(1+M_f) + M_mM_f(1+A_v)} \end{aligned} \tag{4}$$

Sizing is iterative using a simple bisection algorithm, including a boundary detection and extension mechanism as shown in fig.10. This application converged systematically in under a second (typically a few tens of iterations) to a precision of better than 0.01% on a Sun Ultra 5 workstation. The desired TIA performance criteria (transimpedance gain Z_g, bandwidth BW_e and quality factor Q) and operating conditions (photodiode capacitance C_d and load capacitance C_l) allow the generation of component values for the feedback resistance R_f and the voltage amplifier (open loop gain A_v, output resistance R_o).

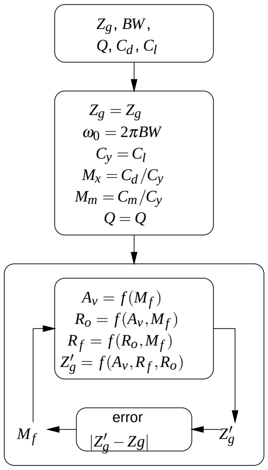


Figure 10. Design procedure for TIA

3.2 Bottom-Up verification methodology

The methodology used for automating the specification verification and correction is shown in fig.8. This is based on the simulation of the complete netlist of the optical receiver, plus the transimpedance architecture. In practice, this is achieved by the following equation, applied to each performance criterion: $S_{corr} = S_{old} \pm \Delta$, where $\Delta = P_{req} - P_{sim}$ and P_{req} represents the performance requirement reached by behavioral model simulation during the top-down phase; P_{sim} represents the simulated performance value generated during the bottom-up verification phase; S_{old} is the specification corresponding to the performance requirement (P_{req}); and S_{corr} is the corrected specification value to be used in a new sizing process.

The verification of the TIA is based on the simulation of the complete netlist with $Spectre^{TM}$. The physical dimensions are extracted directly before the optimization of the fast inverter as shown in fig.8. If all specifications are satisfied then this hierarchical level is considered to be qualified. Otherwise the specification for the gain of TIA and BW_e are corrected, the new values of the capacitances C_i,C_m, C_o are extracted by the library function and a new evaluation of TIA begins.
Optical receiver performance verification and correction is achieved using simulation of the complete netlist representing the HDL-A photodiode model [5] and TIA structure with the Eldo simulator [5]. The physical transistor dimensions representing the fast-inverter are extracted directly before the optimization of the TIA has finished. For the photodiode we extracted the final value of performance.

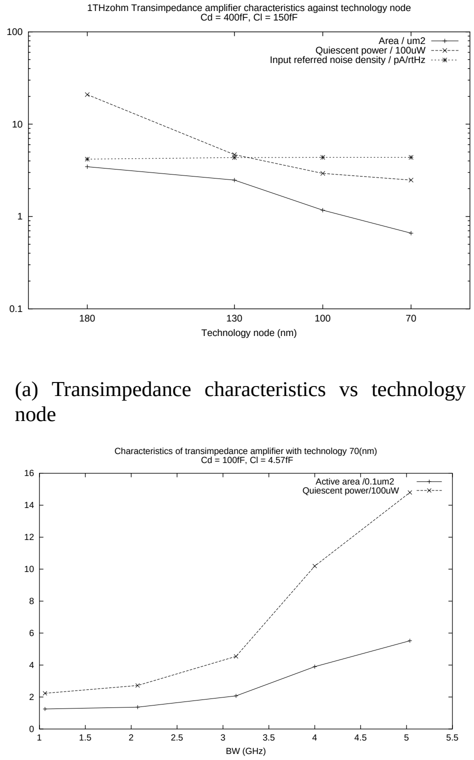
4. Results

As an example of the validation of the described approach allowed by Rune, the method was used to design a 0.13μm CMOS 1$THz\Omega$ TIA with an InGaAs PIN photodiode.

Parameter	Value
Optical power P_l	*50 μW*
Wavelength λ	*850nm*
Bandwidth BW_e	*1.1 GHz*
Junction capacitance C_d	*94.1 fF*
Photocurrent I_{ph}	*42.3 μA*
Photodiode reverse bias voltage V_d	*1.87 V*
Intrinsic zone thickness w_d	*10 μm*
Photodiode responsivity R_q	*0.85 A/W*
Transimpedance gain Z_{g0}	*62.6 db*
M_1 transistor width W_1	*90.4 μm*
M_2 transistor width W_2	*4.2 μm*
M_3 transistor width W_3	*27.0 μm*
M_{1-3} transistor lengths L	*0.13 μm*
Transimpedance feedback resistance R_f	*1.5 k Ω*
Supply voltage V_{dd}	*1.2V*
Load capacitance C_l	*6.47fF*
DC input voltage V_{in}	*0.7 V*
DC output voltage V_{out}	*0.6 V*
Quiescent power	*4.2 mW*

Figure 11. Simulated performance of photoreceiver

The simulated photoreceiver performance is summarized in fig.11. Using the optimization methodology for the TIA with BSIM3v3 models for 0.35um technology and accurate specification shown in fig.13(a), we have demonstrated the efficiency of the RUNE platform. The specification Q allows a large real BW_e, and V_i and V_o are specified as $V_{dd}/2$ allowing maximum gain for the internal fast amplifier. The optimization results and performance characteristics are shown in fig.13(b). For technology nodes from 180nm down to 70nm [2],we also generated design parameters for 1THzΩ TIAs to evaluate the evolution in critical characteristics with technology node.



(a) Transimpedance characteristics vs technology node

(b) Transimpedance characteristics vs bandwidth with technology 70nm

Figure 12. Transimpedance characteristics vs technology and bandwidth

Fig.12(a) shows the results of transistor level simulations of fully generated TIA circuits at each technology node. According to traditional *"shrink"* predictions, which consider the effect of applying a unitless scale factor of $1/k$ to the geometry of MOS transistors, the quiescent power and device area should decrease with $1/k^2$ factor. Between 180nm and 70nm technology nodes $k^2 \simeq 6.61$, which is verified through our sizing optimization procedure. And finally with this methodology we can find a particular specification to a given tolerance, as shown in fig.12(b). We have plotted the active area of the generated TIA with static power dissipation for bandwidths 1GHz to 5GHz with Z_g at 1kΩ and the quality factor Q at $1/\sqrt{2}$.

5. Conclusion

In this paper, a tool for co-synthesizable analog and multi-domain IP, was presented. The framework has been developed to exploit the IP blocks thus formalized in an entirely configurable association of encapsulated design methodologies with heterogeneous evaluation tools.

Condition		Specification	Tolerance	Result
technology (nm)	350	BW_e (GHz)= 1.5	0.05%	1.473
$V_{dd}(V)$	3.5	$Z_g(K\Omega) = 1$	0.02%	1.006
$C_l(fF)$	150	pwr (mW)	–	6.12
$C_d(fF)$	400	C_m (fF)	–	17.43
$I_d(uA)$	50	C_i (fF)	–	45.503
		$R_f(K\Omega)$	–	1.406
		$V_i(V)$= 1.65	0.05%	1.58
		$V_o(V)$= 1.65	0.02%	1.62
		$Q = 1/\sqrt{2}$	0.004%	0.7045

(a) Transimpedance specification and result

Transistor size	
All transistor'L (um)	0.35
W(M1)/L	11.55
W(M2)/L	50.53
W(M3)/L	2.712

CPU Characteristic	
time	30 mn
Bottom-up loop number	6
$Spectre^{TM}$simulation number	265

(b) Results of transistor sizing and CPU characteristics

Figure 13. Optimization characteristics and results of 0.35um CMOS TIA

The framework and IP model have been used successfully in the design of a high-speed integrated optoelectronic photoreceiver with accurate performance. The results of this methodology for a high speed CMOS photoreceiver have been used to provide an objective and quantitative comparison between electrical and optical clock distribution networks in terms of dissipated power.

References

[1] Z. B. Wilson and I. Darwazeh. Analogue optical fiber communication. *IEE Press, U.K.*, October 1995.

[2] Y. Cao, T. Sato, D. Sylvester, M. Orchansky, and C. Hu. New paradigm of predictive MOSFET and interconnect modeling for early circuit design. In *Proc. Custom Integrated Circuit Conference*, January 2000.

[3] J. Graeme. *Photodiode Amplifiers*. McGraw-Hill, 1996.

[4] M. Ingels and M. S. J. Steyaert. A 1-Gb/s, 0.7μm CMOS optical receiver with full rail-to-rail output swing. *IEEE Journal of Solid-State Circuits*, 34(7), July 1999.

[5] Mentor Graphics Corporation. Analog/Mixed-signal Simulators and Libraries Bookcase, 2003.

[6] T. Mukherjee, G. K. Fedder, and R. S. Blanton. Hierarchical design and test of integrated microsystems. *IEEE Design and Test of Computers*, 16(4), October-December 1999.

[7] I. O'Connor, F. Mieyeville, F. Tissafi-Drissi, G. Tosik, and F. Gaffiot. Predictive design space exploration of maximum bandwith CMOS photoreceiver preamplifiers. In *Proc. IEEE International Conference on Electronis, Circuit and Systems*, December 2003, in press.