Hierarchical synthesis of high-speed CMOS photoreceiver front-ends using a multi-domain behavioural description language

Faress Tissafi-Drissi, Ian O'Connor, Fabien Mieyeville, 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}@ec-lyon.fr

Abstract

We present a hierarchical design methodology and synthesis platform aimed at fast multi-domain system design automation using behavioural models. 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 enables the generation of increasingly complex functions and greater processing power. Efficient evaluation of these circuits requires accurate and meaningful behavioural description levels. Sizing at transistor level becomes prohibitive above a certain number of transistors since the number of design parameters, N, is of the same order of magnitude as the number of transistors in the block. If the number of solutions visited for each individual design parameter is represented by s, the time necessary for each evaluation carried out during the design process by T_e, and the parameter generation time by t_g, then the total design time t_d is

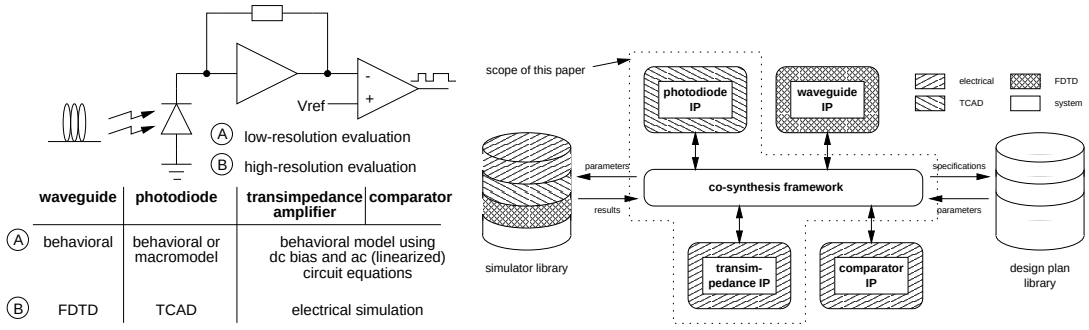
$$t_d = (T_e + t_g) \prod_{i=1}^{N} s_i$$

which is clearly impractical for large blocks. For this reason, we must use techniques such as behavioural modelling to simplify the design problem, and therefore design tools capable of handling design problems over several hierarchical levels are necessary. As concerns heterogeneity, integrated systems are progressively taking on board elements of different nature (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 optimised and the design process is inefficient. Again, multi-domain behavioural models or co-simulation backplanes are key to this issue, but so are design automation tools in order to allow even constraint distribution over elements from different physical domains.

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. Current interest in such objects is due to their position as a potential solution to the interconnect bottleneck predicted for ICs requiring high-frequency (above 5GHz) operation [1]. At the very least, they could be used to replace electrical input/output pads and wire bonds to transfer information on and off the chip [2], but

they could also potentially find uses in clock distribution, across-chip data transport and (using wavelength routing) high-density buses and switch boxes in reconfigurable computing [3]. However, their design using existing tools is difficult since complete link simulation requires digital, analog and photonic simulation engines in a common environment, and design methodologies for such links require different design styles at various hierarchical levels, and range from design space exploration to final design centering and optimization.

To illustrate these last points, fig. 1(a) shows the receiving end of an integrated optical link. The performance of this link can be simulated (A) with parameterised behavioural 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 [4], [5], but in practice this proves difficult and time-consuming. Our solution consists of (i) carrying out a top-down design space exploration using behavioural models to the physical level, (ii) physical sizing linking directly from the co-synthesis backplane to the various evaluation tools, as shown in fig. 1(b), and (iii) subsequent bottom-up design verification using model parameter extraction. Reconfigurable links to design plans allow the user to change design styles to suit the application: for example fast, interactive design space exploration to quantitavely predict the benefits of using novel architectures (and, conversely, establishing device performance objectives such that benefits are obtained); or again, numerical optimization based on detailed simulation to finalise a design.



(a) System schematic

(b) Cosynthesis backplane showing multiple simulators and design plans

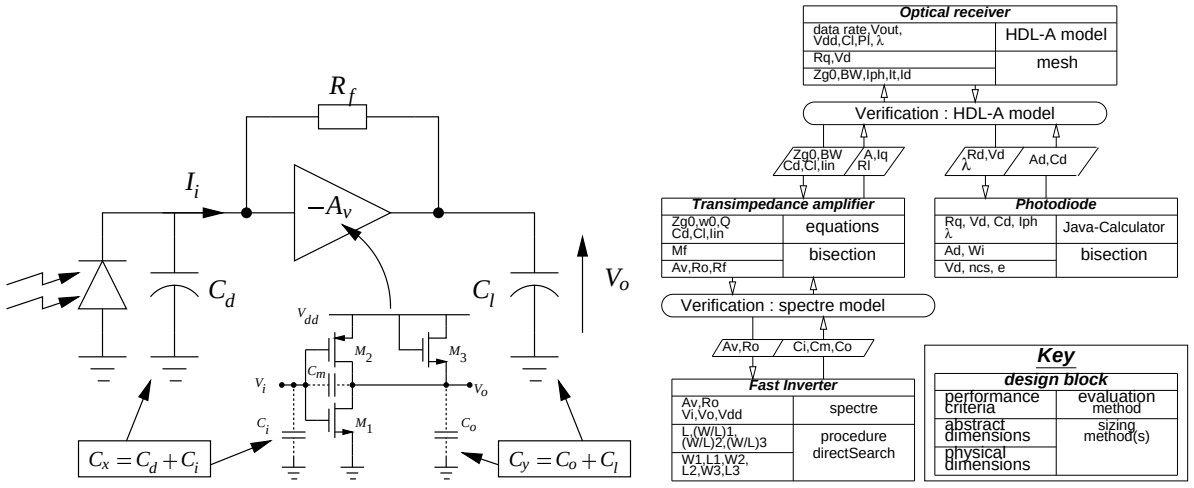
Figure 1. Multi-domain system example: integrated optical receiver

Section 2 describes in more detail the problems involved in the design of high-speed photoreceiver front-ends and gives an overview of the application-specific design flow implemented. Our platform facilitates design methodology implementation for systems integrating devices of different natures. We show in section 3 how we are able to capture this design flow and block information including behavioural models, associating sizing algorithms and standard evaluation tools. Section 4 describes the top-down, physical and bottom-up design phases involved. Finally in section 5, to illustrate the use of the framework, we describe the performance of the described approach in the design of a high-speed CMOS photoreceiver front-end.

2. Photoreceiver design flow

High-speed CMOS photoreceivers 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. Design

through reasoned hierarchical decomposition, already difficult for all-electrical systems, takes on an extra dimension when considering systems integrating blocks of different physical natures (optical, electronic, mechanical ...). While the architecture may be known, a balanced parameterisation of each block is difficult to achieve since dedicated, domain-specific simulators allow only a very rudimentary analysis of the interface, usually through equivalent first order models. Behavioural languages such as HDL-A, VHDL-AMS or Verilog-A allow modelling of physical behaviour from different physical domains, and simulators such as Spectre, Eldo, AdvanceMS or Hamster then allow concurrent simulation of such models, independent of their type. Our platform, through its generic evaluation interface, is capable of using multi-domain behavioural model simulation in order to optimise heterogeneous structures. In this application, we have used HDL-A models with the Eldo simulator.



(a) Photoreceiver front-end structure

(b) Flow model for receiver synthesis

Figure 2. Photoreceiver structure and design flow model

The basic photoreceiver structure in a typical configuration is shown in fig. 2(a) [6]. The internal amplifier is a single-ended inverter structure with diode load to increase its bandwidth. The main limit to high-speed operation is often the photodiode capacitance C_d. Typically, the design flow of the photoreceiver as a whole is fragmented at the photodiode/transimpedance interface, as the photodiode is designed first and circuit design "suffers the consequences" of design choices made at this level. In our design flow (fig. 2(b)), we simultaneously apply design methodologies for both sides in order to evenly distribute design constraints over the various components.

3. Platform architecture

Some necessary stages are 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

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

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. A simplified UML[2] model representing the class structure of the platform is shown in fig. 3.

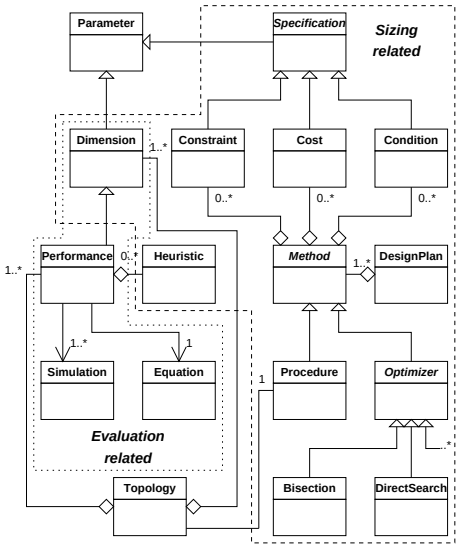


Figure 3. Simplified UML model for platform architecture

In this section, we will first describe the design approach for one hierarchical level. The second part will detail hierarchy management.

3.1 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. 4.

The *topology* object 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 behavioural model and formulated in terms of the physical dimensions of the topology, or (ii) a link to a numerical simulation harness common to all topologies of one type (*category*), instantiating the topology under certain test conditions and targeting specific analyses. For simple performance criteria, the user can also capture an analytical equation in C-like code.
- 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.

[2]Unified Modeling Language

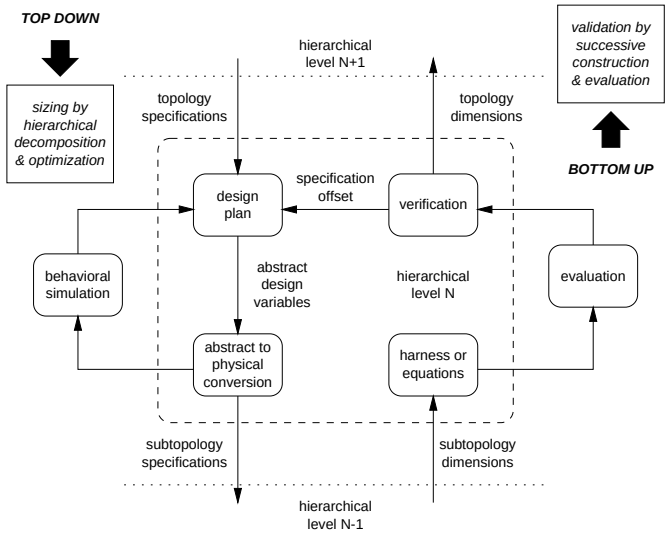
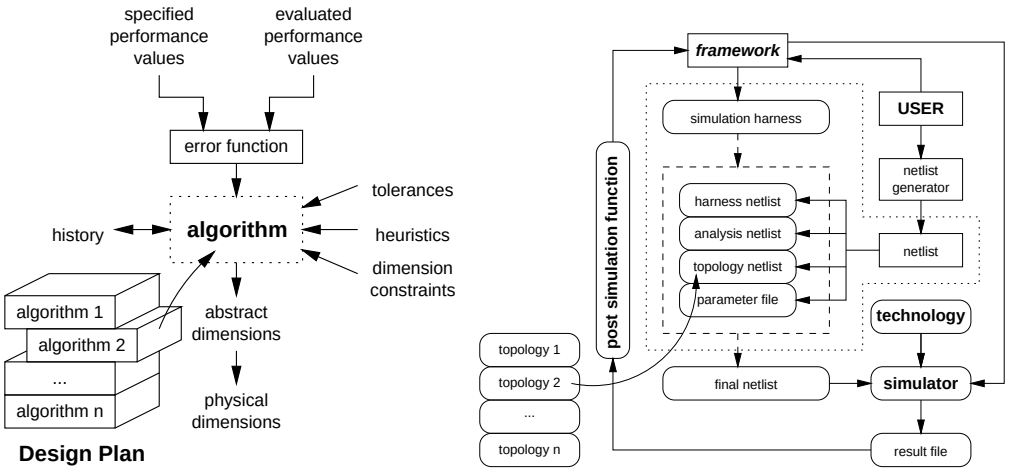


Figure 4. Synthesis flow at one hierarchical level

The manner in which the elements in the topology IP are exploited during the design process is formalised by a *design plan*, representing a sequence or a loop of sizing methods. The capability of drawing on a library of homogenised 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. 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. 5(a) shows what happens



(a) Design plan concept

(b) Simulation harness and interface

Figure 5. Design plan and evaluation interfaces in the platform

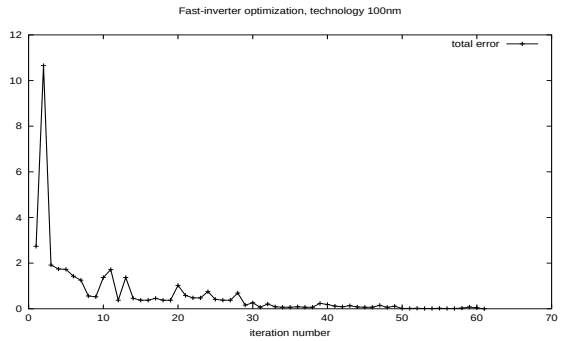
between performance evaluation for one set of dimensions, and generation of the following set. The error function is computed from comparison between specified and evaluated performance values, depending also on the type of specification. The current algorithm in the design plan stack is called for a method "hit" (one iteration) based on the algorithm's tolerances, design history and constraints and (according to

user needs) heuristics. A new set of abstract dimensions is generated and translated into physical dimensions for evaluation. All sizing method classes inherit from an abstract[3] class encapsulating the "black box" requirements for a method to be able to operate within the platform.

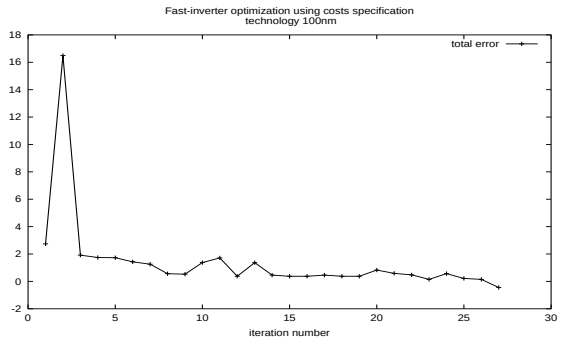
The design objective function itself is built up from a summing of individual performance criteria error functions, of which there can be three types. In the following definitions, ε represents the individual error function contribution of the particular specification, W_i represents the weighting function, P_s the specified performance value and P_r the realised performance value.

$\hookrightarrow$	*Constraints*	*Conditions*	*Costs*	*Error function*	*Result*	
A_v	≥ 7			$\varepsilon_{cs}(1,7,P_r,\geq)$	$\hookrightarrow$	7.32
			maximize, 7	$\varepsilon_{ct}(1,7,P_r,maximize)$	14.462	
$R_o(K\Omega)$	$\leq 1K$			$\varepsilon_{cs}(1,1K,P_r,\leq)$	0.360K	0.221K
$Pwr(mW)$	$\leq 4.5m$			$\varepsilon_{cs}(1,4.5m,P_r,\leq)$	6.238m	4.308m
F_i		$= 1.0$		$\varepsilon_{cd}(10,1,P_r,=)$	1.617	1.005
$V_o(V)$		$= 0.5V$		$\varepsilon_{cd}(1,0.5,P_r,=)$	0.562	0.472

(a) Fast-inverter specification



(b) Fast-inverter optimization without costs



(c) Fast-inverter optimization with costs

Figure 6. Fast-inverter specification and optimization

- 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 minimised. Here $\varepsilon_{ct} = \pm W_i\frac{P_s-P_r}{P_s}$ depending on the type of the cost (maximise or minimise).
- conditions (equalities) which represent fixed points with tolerances. If the real value is outside the tolerances, then $\varepsilon_{cd} = W_i|\frac{P_s-P_r}{P_s}|$.

In table 6(a), the notation ε_c means $\varepsilon_c(W_i,P_s,P_r,ST)$ where ST is the specification type. For example, consider the optimization of the fast-inverter with the specifications shown in fig. 6(a), where A_v is the voltage gain, R_o is the output resistance, Pwr is the quiescent power, V_o is the quiescent output voltage and F_i is a formulation of Kirchoff's current law. Each specification's contribution to the overall error function is shown in fig. 6(a). The total error of this optimization problem is $\varepsilon_t = \varepsilon_x(A_v) + \varepsilon_{cs}(R_o) + \varepsilon_{cs}(Pwr) + \varepsilon_{cd}(F_i) + \varepsilon_{cd}(V_o)$. The optimization aim is to minimise the value of the overall error function. In the case where $\varepsilon_x = \varepsilon_{cs}$ the ideal solution is $\varepsilon_t = 0$, and in the other case, where the cost specification for the gain ($\varepsilon_x = \varepsilon_{ct}$) is used, the solution is obtained for negative values of ε_t. For F_i we have shown a condition specification since the significant interval value is more restricted around one value. We have

[3]object-oriented terminology here: a class defining method prototypes

used a direct search algorithm (Hooke and Jeeves) to resolve this example. The reduction in total error with the number of loop iterations in both cases is shown in fig. 6(b) and fig. 6(c), and the optimised result in fig. 6(a), where the left-hand results column shows the fast-inverter optimization result with cost-type gain specification and the right-hand column with constraint-type gain specification.

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 run-time 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(b). 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 standardised tabular form by a simulator-dependent interface. Generation of the simulated performance value is then simply carried out by calling the necessary function from the post-simulation function library.

3.2 Hierarchy management

Our platform processes hierarchically structured systems in a simple and efficient way. This is achieved through the application of the following principles:

- Decomposing a hierarchical system: The first step to be taken is the decomposition of the system into its constituent blocks. Particular care must be taken with respect to the relative environment of the block to be optimised: loads, interactions and common nodes must be accurately modelled and imposed as operating conditions for the block being synthesised.
- Describing a hierarchical system: A hierarchical system requires sub-blocks down to the physical level in order to be designed completely. Component libraries must therefore be populated by building up from the lowest (physical) level. We can also associate individual design plans with each block at any hierarchical level, defining specification tolerances.
- Specification generation: The specifications and process technology for the entire structure to be optimised are defined at the top level. Specifications for each sub-block are generated dynamically during the sizing process. Sizing methods can then vary block parameters until a suitable solution has been found, at which point the blocks must be sized individually using the fixed parameters as specifications. During decomposition therefore, we transform constraint type parameters into fixed conditions in order to avoid insufficient block performance at the system level. Parameters (performance criteria or dimensions) used for each block can also be propagated up the hierarchy in order to (i) update physically linked parameters (power dissipation, parasitics, ...) and (ii) allow accurate bottom-up verification.
- Bottom-up verification: After successful optimization at a physical level, the platform transmits the obtained performance values to a higher level, which allows the optimization process to continue synthesis with relatively accurate information about real block performance. Once all sub-blocks have been sized, the platform evalutes the performance of the assembled blocks together in order to compare with required performance values. If the verification is successful to within a user-defined tolerance, then the process continues up the hierarchical structure. If not, the initial

specifications are corrected to take into account the error between the specified and generated solution by in effect moving the design space. In practice, this is achieved by the following equation, applied to each performance criterion: $\Delta = P_{req} - P_{sim}$ and $S_{corr} = S_{old} \pm \Delta$. where P_{req} represents the performance requirement reached by behavioural 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.

4. Photoreceiver design methodology

We now present the implementation of 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_o, and generates a current I_{ph} according to its photoresponsivity R_q. 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. The final flow model previously shown in fig. 2(b) uses four blocks at three hierarchical levels.

4.1 Optical receiver

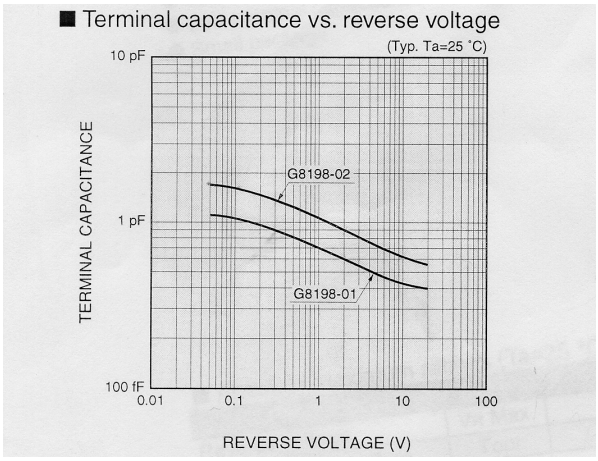
The specifications used at this level are the data rate D, optical input power P_o, wavelength λ, supply voltage V_{dd} and load capacitance C_l. The design parameters concern the photodiode (responsivity R_q, bias voltage V_d, optical bandwidth BW_{opt} and diode capacitance C_d) and the TIA (operating voltages V_i, V_o V_{dd}, transimpedance gain Z_g and electrical bandwidth BW_e).

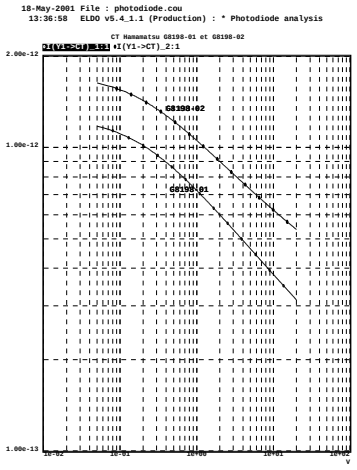
We have used an HDL-A behavioural model for the photodiode, shown in fig. 8. This model has been validated at both static and dynamic levels with experimental measurements of high-speed InGaAs photodiodes from Hamamatsu, giving an error of less than 5%. Fig. 11 shows a comparison of results from simulation of the model and datsheet curves. The behavioural model of the TIA is at this level a simple linear transfer function: $v_{out} = \frac{Z_g}{1+\frac{s}{2\pi BW_e}} v_{in}$

Both behavioural models are included in an Eldo netlist, shown in fig. 8 and in which the simulation harness (jig) is also described. Care must be taken to constrain the sizing problem correctly: for example, the photodiode reverse bias voltage as too small a limit will result in overconstraint and the process will not converge; but too high a limit will place severe constraints on the operating region of the photodiode. We constrained $V_d < 2V_{dd}$, supposing separate supply voltages to the TIA and photodiode. The simulation using the behavioural models carries out a frequential analysis, modelling the optical signal by a voltage source with amplitude equal to that of injected optical power. Bandwidth is extracted at -3dB using the post-processing function library, and extrapolated to the data rate using $D \approx 1.4BW$ to retrieve sufficient signal power above the fundamental. The block parameters are transmitted from the platform to the simulator through the file “parameters.lst”, as shown in fig. 8.

4.2 PIN photodiode

In order to evaluate the photodiode performance during the physical sizing process, we used an internally developed calculator shown in fig. 9, which is based on standard PIN photodiode equations from the literature [7]. At this level, the specifications are the photoresponsivity R_q, junction capacitance C_d, photocurrent I_{ph}, wavelength λ, optical bandwidth BW_{opt}, resistive load R_L and reverse bias voltage V_d. We also define material parameters such as energy gap, absorption coefficient at required wavelength, average carrier mobility, etc. The physical dimensions to be used in the sizing process represent the diode structure: intrinsic zone thickness, area.





(a) Datasheet measurement (b) Simulation results

Figure 7. Comparison of DC simulation and measurements

```
* ATI_PHOTODIODE analysis
.lib /cad/netlist/parameters.lst
.model ATI_PHOTODIODE(physique) macro lang=hdla
.hdlalib /cad/eldo_ATI/model/opto_lib
y1 ATI_PHOTODIODE(physique)
+ pin :  P 3 1
+ coupling :  y1->Cfinale y1->Vdiode y1->Itotal
y1->Iphoto y1->Idiode
+ generic:fc=3.0e+9 Resp=0.85 Rs=50 Rshunt=10e9
RL=50 m=0.33 Vbi=0.20
Isat=0.06e-9 Vref=2 Vin=0.65 Vout=0.65
v2 1 0 DC 2.2
v1 P 0 DC 35u AC 50u
*Transimpedance
.include /cad/Tech_st120nm/corners
XPMOS_1 VOUT VIN VDD VDD EPHSGP_BS3JU
w=+1.00000000E-05 l=+1.20000000E-07 +nfing=1
ncrsd=1.0 number=1.0 srcefirst=1 mismatch=1
XNMOS_2 VDD VDD VOUT VOUT ENHSGP_BS3JU
w=+1.00000000E-05 l=+1.20000000E-07 +nfing=1
ncrsd=1.0 number=1.0 srcefirst=1 mismatch=1
XNMOS_1 VOUT VIN 0 0 ENHSGP_BS3JU
w=+1.00000000E-05 l=+1.20000000E-07 +nfing=1
ncrsd=1.0 number=1.0 srcefirst=1 mismatch=1
*harness photodiode
.connect 3 VIN
*harness transimpedance
Rf VIN VOUT 1.467k
Cout VOUT 0 4.57p
Vdc VDD 0 DC 1.2
*dc analysis
.DC v2 0.05 5 0.01
*ac analysis
.AC dec 100 1 10e9
.end
```

```
ENTITY ATI_PHOTODIODE IS
GENERIC (fc,Resp,Cj0,Rs,
Rshunt,RL,m,Vbi,Isat,Vref,Vin,Vout,d:real);
COUPLING (Cfi-
nale,Vdiode,Itotal,Iphoto,Idiode:analog);
PIN (Popt:nkn ; p1,p2:electrical);
END ENTITY ATI_PHOTODIODE;
ARCHITECTURE physique OF ATI_PHOTODIODE IS
VARIABLE
q,ne,h,c,k,T,nu,Iph0,nideal,Test,nusat:real;
STATE Tau_r,Tau_opt,Tau_elec,Iph,Itot,Cj,Id,ii,vv,Ij,Ishunt,Vd,Expr:analog;
BEGIN
RELATION
PROCEDURAL FOR init =>
...
physical constants
...
PROCEDURAL FOR dc, ac, transient =>
Iph0:=Resp*Real(Popt.a);
vv:=[p1,p2].v;
ii := Id+Ishunt+Ij-Iph;
Itot := ii;
p2.i %= -ii;
p1.i %= ii;
IF Vd<=0.0 then Expr:=(exp(m*ln(1.0-Vd/Vbi)));
Test:=1.0;
ELSE Expr:=1.0/(1.0+m*Vd/Vbi);Test:=0.0;
END IF;
Cj := Cj0/Expr;
Cfinale := Cj;
Vdiode := Vd;
Itotal :=Itot;
Idiode := Id ;
Tau_elec := 2.2*(Rs+RL)*Cj;
Tau_r := 0.35/fc;
EQUATION (Iph, Id, Ij, Ishunt,Vd,Iphoto) for
DC, AC, TRANSIENT =>
Resp*Popt.a == Iph +Tau_opt/2.2*ddt(Iph);
Id == Isat*(exp(q*Vd/(nideal*k*T))-1.0);
Ij == Cj*ddt(Vd);
Ishunt == Vd/Rshunt;
Vd + Rs*(Id+Ishunt+Ij-Iph) == vv;
Iphoto == Iph;
END RELATION;
END ARCHITECTURE physique;
```

Figure 8. Photoreceiver netlist and HDL-A model

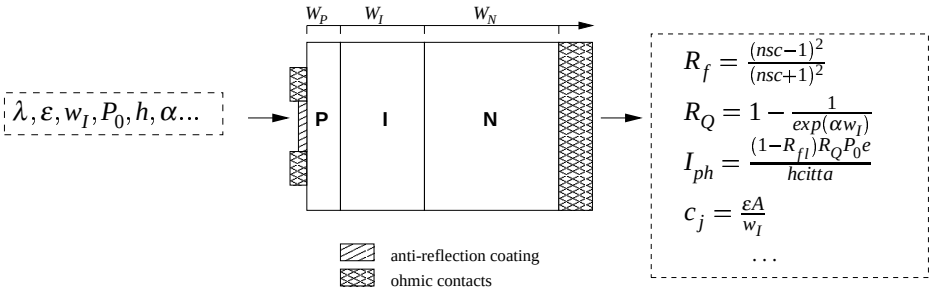


Figure 9. Photodiode calculator

The input impedance of the TIA is simulated by the load resistance R_L. At the beginning of the sizing process, its value is estimated according to TIA specification parameters ($Z_{in} \approx R_f/A_v$). During later iterations, its value is extracted directly from transistor level simulations.

4.3 TIA

The method developed for the TIA, shown in fig. 10(a), is based on a frequency analysis of the structure and a mapping of the component values to coefficients in a Butterworth type filter approximation function [8]. Sizing is iterative using a simple bisection algorithm, including a boundary detection and extension mechanism. 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 generation of component values for the feedback resistance R_f and the voltage amplifer (open loop gain A_v, output resistance R_o). We introduce the multiplying factors $M_f = R_f/R_o$, $M_x = C_x/C_y$ and $M_m = C_m/C_y$, normalising all expressions to $\tau = R_o C_y$.

Vary M_f by bisection and calculate	
$A_v = \frac{a-2bcQ^2 \pm \sqrt{a^2-4abcQ^2}}{2c^2Q^2} - 1$	if $M_m \neq 0$
$A_v = \frac{b^2Q^2}{M_f M_x} - 1$	if $M_m = 0$
$R_o = \frac{\sqrt{\frac{1+A_v}{M_x M_f}}}{\omega_0 C_y}$	
$R_f = M_f R_o$	
$Z'_{g0} = \frac{R_o - R_f A_v}{1+A_v}$	
where: $a = M_f(M_x + M_m + M_x M_m)$	
$b = 1 + M_x(1+M_f)$	
$c = M_m M_f$	
until $\lvert Z_{g0} - Z'_{g0} \rvert < \varepsilon$	

(a) Transimpedance sizing procedure

$V_{gst1} = V_i - V_{t1}$,	$V_{gst2} = V_i - V_{dd} - V_{t2}$
$g_{m1} = \frac{A_v/R_o}{\frac{V_{gst1}}{\lvert V_{gst2} \rvert}+1}$,	$\left(\frac{W}{L}\right)_1 = \frac{g_{m1}}{K_n V_{gst1}}$
$g_{m2} = g_{m1}\frac{V_{gst1}}{\lvert V_{gst2} \rvert}$,	$\left(\frac{W}{L}\right)_2 = \frac{g_{m2}}{K_p \lvert V_{gst2} \rvert}$
$L_{1,2,3} = L_{min}$,	$\left(\frac{W}{L}\right)_3 = \frac{W_{min}}{L_{min}}$

(b) Internal amplifier sizing procedure

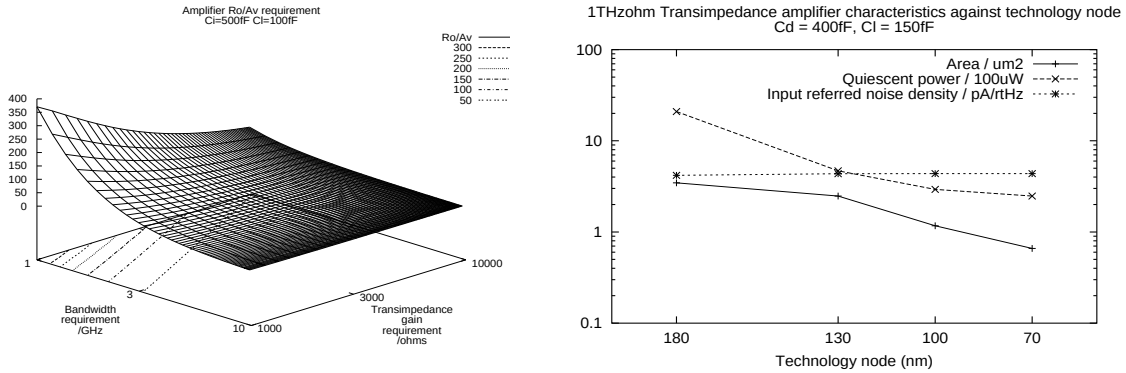
Figure 10. Algorithms for transimpedance sizing

The input impedance of the TIA is extracted at the photoreceiver level to be imposed as condition to photodiode sizing.

4.4 Internal amplifier

At the physical level, the internal amplifier is sized using a dedicated procedure (fig. 10(b)) followed by a direct search optimization algorithm for fine-tuning. The procedure was developed using approximate equations for the small-signal characteristics and bias conditions of the circuit (we supposed $V_{in} = V_{out} = V_{dd}/2$, which gives stable results). During the optimization process, the circuit is evaluated by simulation for exact results. Specifications are simply voltage gain A_v and output resistance R_o; design parameters are the physical transistor sizes. The parasitic capitances are extracted at the TIA level for use in the TIA sizing procedure.

Taking into consideration the physical realisation of the amplifier, those with requirements for low gain and high output resistance (high R_o/A_v ratio) are the easiest to build, and also require the least quiescent current and area. We plot this ratio against bandwidth and transimpedance gain requirements (fig. 11(a)).



(a) Design space Ro/gain requirements

(b) Transimpedance characteristics vs technology node

Figure 11. Transimpedance exploration design and characteristics vs technology node

5. Results

As an example of the type of validation of the described approach, the method was used to design a 0.13μm CMOS $2THz\Omega$ TIA with an InGaAs PIN photodiode. The simulated photoreceiver performance is summarised in table 1.

Using this methodology and predictive BSIM3v3 models for technology nodes from 180nm down to 70nm [9], we also generated design parameters for $1THz\Omega$ TIAs to evaluate the evolution in critical characteristics with technology node. Fig. 11(b) shows the results of transistor level simulation of fully generated TIA circuits at each technology node.

6. Conclusion

We have presented the implementation of a complete hierarchical synthesis methodology for high-speed photoreceiver front-ends. A framework has been described, developed to exploit multi-domain IP blocks in an entirely configurable association of encapsulated design methodologies with heterogeneous evaluation tools. A particularly crucial point is the importance of behavioural modelling for hierarchical synthesis. We have developed a method of communicating between different hierarchical levels based on such models.

Table 1. Simulated performance of photoreceiver

Parameter	Value
Optical power P_{opt}	*50* μW
Wavelength λ	*850nm*
Bandwidth BW	*1.1 GHz*
Junction capacitance C_j	*94.1 fF*
Photocurrent I_{ph}	*42.3* μA
Photodiode reverse bias voltage V_d	*1.87 V*
Intrinsic zone thickness d	*10* μm
Photodiode responsivity R	*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* $k\,\Omega$
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*

References

[1] D. A. B. Miller, "Physical reasons for optical interconnections," *Int. J. Optoelectronics*, vol. 11, 1997.

[2] K. W. Goossen, "Optoelectronic/VLSI," *IEEE Trans. on Advanced Packaging*, vol. 22, no. 4, Nov. 1999.

[3] N. Savage, "Linking with light [high-speed optical interconnects]," *IEEE Spectrum*, vol. 39, no. 8, Aug. 2002.

[4] F. Gaffiot, K. Vuorinen, F. Mieyeville *et al.*, "Behavioral modeling for hierarchical simulation of optronic systems," *IEEE Trans. Circuits and Systems II*, vol. 46, no. 10, Oct. 1999.

[5] T. Mukherjee, G. K. Fedder, and R. S. Blanton, "Hierarchical design and test of integrated microsystems," *IEEE Design and Test of Computers*, vol. 16, no. 4, October-December 1999.

[6] 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*, vol. 34, no. 7, July 1999.

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

[8] I. O'Connor, F. Mieyeville, F. Tissafi-Drissi *et al.*, "Exploration paramétrique d'amplificateurs de transimpédance CMOS à bande passante maximisée," in *Proc. Colloque Traitement Analogique de l'Information, du Signal et ses Applications*, 2002.

[9] Y. Cao, T. Sato, D. Sylvester *et al.*, "New paradigm of predictive MOSFET and interconnect modeling for early circuit design," in *Proc. Custom Integrated Circuit Conference*, June 2000.