High-Level Virtual Prototyping of Signal Integrity in Bus Communication

Ruomin Wang, Julien Denoulet, Sylvain Feruglio, Farouk Vallette, Patrick Garda

High Level Virtual Prototyping of Signal Integrity in Buses Communication

Ruomin Wang, Julien Denoulet, Sylvain Feruglio, *Member, IEEE*, Farouk Vallette, and Patrick Garda, *Senior Member, IEEE*

***Abstract* — In this paper, a novel methodology for high-level modeling of bus communication in embedded systems is introduced. It allows dynamic evaluation of their signal integrity characteristics at the virtual prototyping step (i.e. before physical realization). The method is based on the association of functional and non-functional modules. Functional modules represent the ideal behavior of the system, while non-functional modules use neural networks to model signal integrity effects. This approach was implemented in SystemC-AMS, using the timed data flow model of computation. The method is illustrated by an USB 3.0 application, where modular and parameterizable models are introduced. The method achieved good accuracy (<5 %) while allowing significant simulation speedup (up to ×2000), compared to SPICE-based reference models. This methodology can be used to perform early signal integrity analysis in the virtual prototyping of bus communication in embedded systems.**

***Index Terms*— bus communication, embedded systems, high level, non-functional, modeling, neural network, signal integrity, simulation, SystemC-AMS, system level, timed data flow, virtual prototyping.**

I. Introduction

The ongoing advances of technology have profoundly transformed the potential of electronic chips in the last decades. By following Moore's Law, integration capabilities now allow the design of billion-transistor circuits. Also, heterogeneous association of digital and analog components, hardware and software functions on the same chip, has been a major evolution in the past few years, with the rise of system-on-chips. These developments gave engineers the opportunity to create always more complex applications.

Design methodology of these systems is a key issue for the electronics industry [1]. A way to address this matter and keep up with integration improvements is to increase the design abstraction level from the circuit to the system level [2]. Thus, design flow typically starts by the creation of a high-level model (or virtual prototype) of the system in languages, such as SystemC, that can handle this abstraction level. IP reuse is another crucial element to optimize the development of virtual prototypes, so models of basic blocks for such platforms (processors, memories, interconnects, peripherals …) are often gathered in component libraries. This "top-down" approach can be found in a number of design tools such as Mentor Graphics' Vista, Synopsys's CoMET-METeor, or Open Virtual Platforms. The use of high-level simulation models allows fast virtual prototyping of complex applications (for instance multiprocessor architecture running embedded software and interacting with specialized co-processors). However, the lack of knowledge of the hardware platform makes it difficult to add technological parameters such as power consumption or Signal Integrity (SI) in the simulation process. As a result, in most cases, the simulation of such platforms is ideal.

Indeed, some crucial issues (such as coupling between analog and digital functions, crosstalk noise in on-chip interconnects [3-4], or unexpected hazards caused by low-level effects [5-6]) are often only addressed at the low-level simulation step, even sometimes at the prototyping step, as part of a "bottom-up" design methodology. In that case, the hardware platform is usually well known, so models can include detailed technological parameters. However, the low abstraction level of the models is a real handicap to efficiently simulate complex applications running on large systems, such as the ones described in the previous paragraph.

In the literature, signal integrity effects for high-speed interconnects or transmission lines are modeled using different techniques, as presented in [7]. Among those different methods, we can list RLC and Partial Element Equivalent Circuit (PEEC) models [8-9], piecewise linear models [10], methods based on Finite Difference Time Domain (FDTD) [11], Traveling-wave-based Waveform Approximation (TWA) [12], or models derived from empirical methods [13-14]. All these models are at the circuit level. They aim at representing the signal integrity performances of a single component (a chip, a transmission line). Although simulation speed can be an issue in some cases [15-16], these models cannot be used in a global system-level simulation, to analyze for instance the software/hardware interactions in a heterogeneous system.

Our goal is to enrich the characteristics of high-level virtual prototyping tools by adding to the ideal functional models a performance model. Such tools would help system designers to detect SI issues at an early stage of the design process.

In [17-18], we introduced a meet-in-the-middle modeling approach [19] to evaluate the signal integrity characteristics of field bus-based systems at the virtual prototyping step. This methodology combines high-level SystemC [20] functional

Manuscript received October 9th, 2015.

Authors are with Sorbonne Universités, UPMC Univ Paris 06, CNRS, UMR 7606, Laboratoire d'Informatique de Paris 6 (LIP6), F-75005, Paris, France (e-mail: firstname.lastname@upmc.fr).

models of the nodes, with circuit-level SystemC-AMS [21] models of the I/O and the bus lines. This method allowed global system simulation, while also accurately illustrating low level effects, such as crosstalk between bus lines or the influence of a chip's activity on its power rails' voltage. However, its simulation speed still suffered from the low abstraction level of I/O models, and the method's lack of flexibility made it unsuitable for a virtual prototyping library.

In this paper, a novel modeling approach is proposed, which raises the abstraction of models (such as I/O and transmission lines) to the system level, in order to achieve efficient simulation performances. The method also allows modularity and parameterization of the models, which would make it suitable for use in a virtual prototyping tool. For in this context, a system designer usually selects basic components to model its application then configures these modules' parameters. To do so, neural networks are used to incorporate signal integrity effects in a system level model of a component.

The paper is structured in five parts. Section II presents the modeling methodology based on the association of functional and non-functional modules. Section III focuses on the design of neural-network-based non-functional models and is illustrated by an I²C platform use case. Section IV presents an application based on an USB 3.0 transceiver, and shows how the method could be used in a virtual prototyping library context. Finally, section V concludes the article.

II. Modeling Methodology

In this section, the virtual prototyping method to simulate a system's SI performances along with its functionality is presented. All the models presented in the next sections were developed in SystemC and its analog extensions SystemC-AMS. These sets of open-source C++ libraries offer a unified system-level modeling environment to design and simulate heterogeneous applications, from the processors embedded software to the analog components of a system. It allows modeling at higher levels of abstraction, to improve simulation performance (speed) and efficiency. SystemC operates with a discrete-event simulation kernel, whereas SystemC-AMS features three Models of Computation (MoC). These three kernels allow AMS behavioral modeling at different levels of abstraction, from the more abstract, discrete-time sampled TDF (Timed Data Flow) MoC to the continuous-time and conservative ELN (Electrical Linear Networks) MoC, which is slower than TDF. We chose TDF for our models in order to achieve maximum simulation speed.

With this methodology, a generic bus communication system as shown in Fig.1 will be modeled with two kinds of blocks: functional modules, which represent the operating behavior of the system, and non-functional modules, which manage the SI performances (Fig.2).

A. Functional modules

Functional modules represent the ideal behavior of the system. For example, in the system shown in Fig.1, both nodes' functionalities (such as embedded software), I/O controller and even bus protocol functions (such as I²C wired-and mechanism)

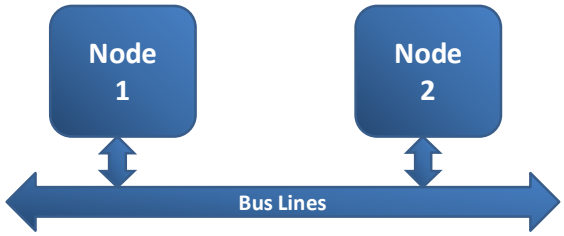


Fig.1. Generic bus system functional model.

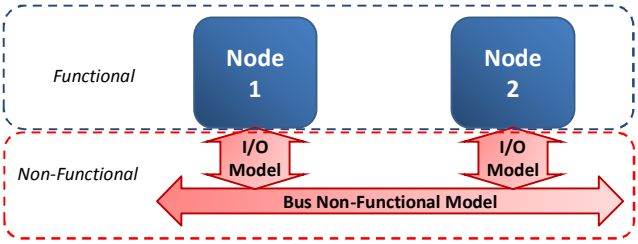


Fig.2. Generic bus system with separation of both functional and non-functional models.

are modeled with these modules. Depending on the nature of the components (software/hardware, digital/analog), C/C++, SystemC and SystemC-AMS languages can be used. Functional models can be independently simulated to visualize the ideal behavior of the bus communication system.

B. Non-functional modules

Non-functional modules are used to represent, at a high-level of abstraction, the system's SI behavior, which is usually highly non-linear. To achieve this, non-functional modules are based on neural networks. As we know, neural networks [22] have been used in a variety of applications [23], because of two important properties: the ability to learn from input data with or without a teacher and the ability to model non-linear functions [24]. Neural networks thus allow to model non-linear SI effects, even with a limited knowledge of the devices. Indeed, at an early design stage, technological features or equivalent circuits may not be available [25]. Based on the application complexity, one neural network can be used for the whole system or one dedicated neural network can be used for each component. Thus, the model can be built as a modular platform. Parameters, such as transceiver configuration or temperature, can also be added to the neural network design, to improve the efficiency and/or flexibility of the non-functional model.

Once neural networks for SI are built, they provide equations that represent the relation between input and output signals. The TDF MoC of SystemC-AMS can efficiently implement these equations. Combined with functional modules, non-functional modules show the SI performances of the system (crosstalk between adjacent bus lines, I/O influence on signal quality, IR drop …). Meanwhile, since these modules can be parameterized, they can help designers optimize their systems (for instance, by trying different transceiver configurations) at the virtual prototyping step.

Since the core of our work is the high-level modeling of SI effects, in the rest of the paper, we mainly focus on the building process of non-functional modules.

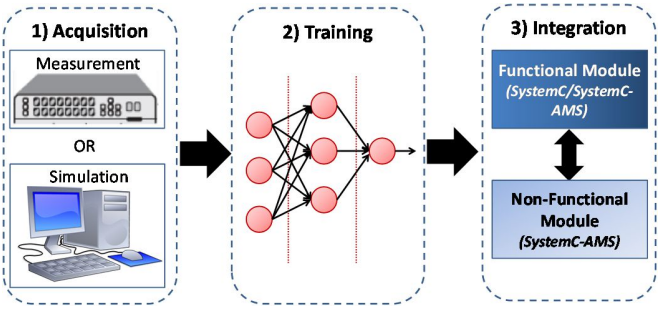


Fig.3. Non-functional module design methodology.

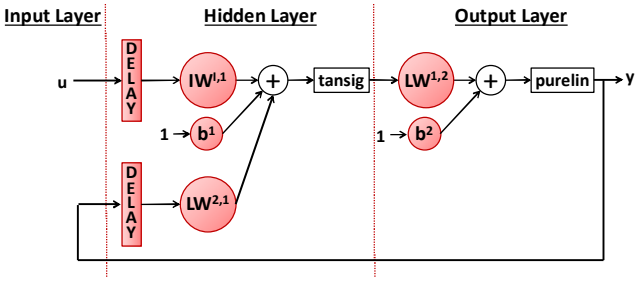


Fig.4. Three-layer neural network with delays and recurrent branch (NARX architecture).

III. NON-FUNCTIONAL MODULE DESIGN

Three stages were required to build non-functional modules, (see Fig.3): first, input/target pairs (such as input voltage/output voltage) were acquired from the devices by measurement or simulation. Then, a neural network architecture was chosen, then trained with the input/target pairs, in order to model the system's non-linear behavior due to SI effects. Finally, the neural network was implemented as a SystemC-AMS TDF module, and was instantiated in the modeling platform. This process is detailed in the rest of this section.

A. Acquisition of input/target signal pairs

In the first stage, (time varying) inputs were given as stimuli to a component (e.g. a transmission line) or a system. These inputs were swept these inputs over their entire operating ranges [26]. (Time varying) outputs were then monitored and stored as targets. These values could be obtained by measurement or simulation.

Input/target signal pairs can be set according to the investigated SI effects. For instance, the most critical SI problem in a two-line bus device could be the crosstalk between adjacent lines A and B. Such a phenomenon typically happens when, for example, a signal transition on line A disturbs the behavior of line B. Hence, input/target signals pairs in this case should focus more on signal transitions than on steady logic levels. However, the combination of several effects can also be investigated if required.

B. Neural network training

In the second stage, neural networks were trained to approximate the relations between inputs and targets. Learning rules of neural networks fall into three major categories (supervised learning, unsupervised learning and reinforcement learning), each of them corresponding to a particular abstract learning task [23], [27]. Supervised learning was used to train the networks. The training was performed with the neural network toolbox of MATLAB [28], which offers a variety of functions based on different architectures of neural networks.

First, a network architecture that was suitable for the studied problem had to be chosen. Then, parameters of the chosen network were set, such as the number of layers of the network, the number of neurons in the hidden layer, the transfer function of each layer.

Unfortunately, there are no predefined rules to help find the best configuration. Depending on the system to be modeled, these configuration parameters should be fixed empirically.

Nevertheless, some guidelines are provided in the literature. For example, in [23], [25], authors indicate that a three-layer network (input layer, hidden layer and output layer) with a sigmoid transfer function (*tansig*) in the hidden layer (1), and a linear function (*purelin*) in the output layer (2), can be trained to efficiently approximate most functions. So this setup was used for our neural networks.

$$tansig(x) = \frac{2}{1+e^{-2x}} - 1 \quad (1)$$

$$purelin(x) = x \quad (2)$$

Fig.4 shows an example of a three-layer network, featuring one input u, one output y, one neuron in the hidden layer, one neuron in the output layer, a recurrent branch between output and hidden layers, and delay functions added for the input and the recurrent branch. $IW^{i,j}$ or $LW^{i,j}$ represent the weight of the connection between neuron i and j, b^i is the bias of neuron i.

The number of neurons in the hidden layer can be chosen according to heuristics, which are typically used as a starting point for a search towards the optimum number. The heuristic $N > 3 \times N_i$ [29] was used, with N the number of neurons in the hidden layer and N_i the number of inputs. Furthermore, the number of training iterations (*epoch*), and, if needed, the number of delay for an input or a recurrent branch have to be set. Delay functions should be present if there is a significant delay between input/target pairs, or if there is a memory effect in the modeled system, such as a capacitance. Large values of N and *epoch* are needed when the system is complex. However, these large values may lead to over-fitting, so a trade-off may be necessary. All of these parameters were fixed by means of empirical testing or trial and error.

The learning rule used in our method was scaled conjugate gradient back propagation (*transcg*) [30], a type of supervised learning that requires less memory. To train the neural network (Fig.5), a subset of the input/target signal pairs obtained during the previous step is used. First the network is fed with the training input data, and its outputs are compared with the training target data (which represent the desired behavior). Following this comparison, the network's weights and biases are updated and the process is iterated *epoch* times. Then, another subset of the input/target signal is used to validate the quality of the training: the error between the validation target data and the network response to the validation input data should be under a predetermined satisfaction threshold.

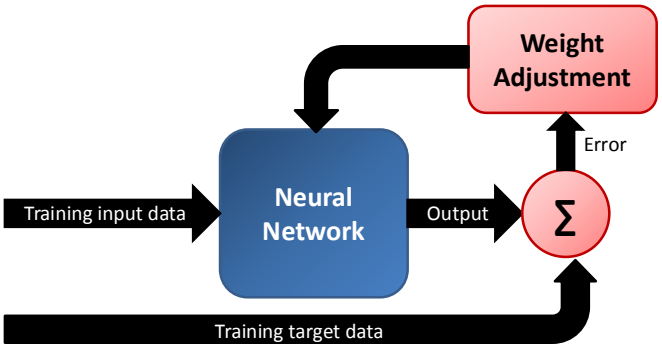


Fig 5. Neural network training

The training process can be time-consuming (from several minutes to several hours). However, it is not necessarily aberrant when one thinks of the time required to design a model in other languages. At the end of this stage, an equation is obtained, which gives an approximation of the relation between the inputs and outputs of the system.

C. *Model Integration*

In the third stage, the neural networks previously generated were implemented into a SystemC-AMS TDF model and were then associated with functional modules. TDF is derived from the well-known Synchronous Data Flow (SDF) model. Unlike the untimed SDF, TDF is a discrete-time modeling style. In a TDF module, a system is described by mathematical functions, (e.g. a transfer function). Therefore, neural networks equations can be easily implemented. SystemC-AMS offers two versions of TDF: *conventional TDF* [21], in which data are sampled with a fixed time step, and *dynamic TDF* [31], in which the time step can be dynamically changed. Compared to a conventional TDF module, a dynamic TDF module samples and computes less data, thus achieving a potential simulation speedup [32]. This feature will be illustrated in the next subsection.

Finally, the SystemC-AMS non-functional model(s) could be connected to the functional module(s) (also written in SystemC/SystemC-AMS) to represent the SI characteristics of the system.

D. *Use case: simulation of an I²C platform*

A use case is now presented to demonstrate the validity of the methodology, the association of functional and non-functional modules, and also the contribution of dynamic TDF to the simulation performance. To do so, a two-node I²C application [33] was used, as shown in Fig.6. It was previously modeled in [18] with SystemC-AMS, but using the ELN MoC (i.e. RLC equivalent circuits). It provided a comparison basis for the TDF/neural-network-based approach.

The application featured a master node (an 8051 microcontroller with a bus controller), and a slave node (an I²C memory device). It was modeled with two functional modules (the microcontroller and the RAM device), and one non-functional module, which incorporated the SI characteristics of both nodes' I/O interfaces and the I²C bus lines (Fig.7). This non-functional module was implemented in conventional TDF, and also in dynamic TDF.

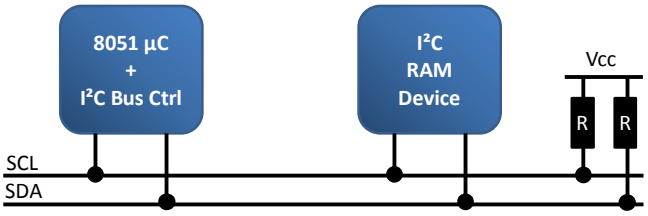


Fig.6. I²C use case.

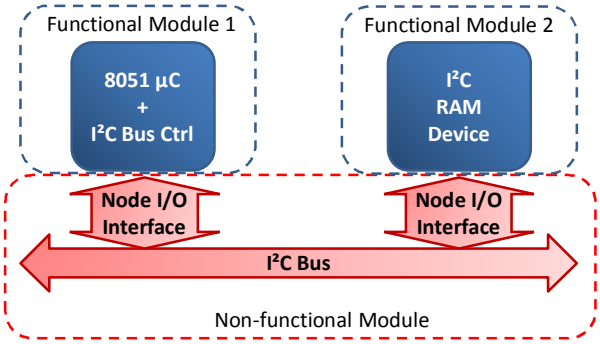


Fig 7. I²C modelling platform.

TABLE I
CONFIGURATION OF NARX NETWORK

D_I	D_R	N	*epoch*	*Training duration*
2	4	10	4500	4h

The SystemC functional modules simulated the execution of the 8051 embedded code, which performed read and write requests to access the I²C memory slave device. The I²C bus controller translated these requests to I²C frames, which were then sent to the bus. For an ideal simulation, a SystemC model of the bus lines could be added to the platform, to implement the wired-and mechanism featured in the I²C protocol. This ideal model would help a designer validate the functionality of the application, but it would not be able to detect any SI effect.

To build the non-functional module, the three-stage process presented above was followed. The first two stages were already introduced in [32]. Input/target pairs were obtained by simulation of an equivalent circuit with Ngspice (Fig.8). This circuit included three blocks: the node's I/O device (featuring a switch and resistor to model the open-drain transistor required by the I²C protocol), an equivalent model of the bus lines (SDA and SCL) [34] and two load resistances. The inputs were the logic levels $V_{IN\text{-}SDA}$ and $V_{IN\text{-}SCL}$. The targets were the output voltages $V_{OUT\text{-}SDA}$ and $V_{OUT\text{-}SCL}$. Around 200000 input/target pairs were obtained, and then used to train the neural network during the second stage. Note that the input/target pairs can also be collected by measurements.

In the second stage, a Non-linear Auto Regressive model with eXogenous (NARX) network was chosen [35]. This recurrent dynamic neural network takes into account the memory effect in the bus RLC equivalent circuit. Its structure is shown in Fig.4. To configure the network, the number of delay for the input (D_I), the number of delay for the recurrent branch (D_R), the number of neurons in the hidden layer (N) and the number of iterations (*epoch*) were set.

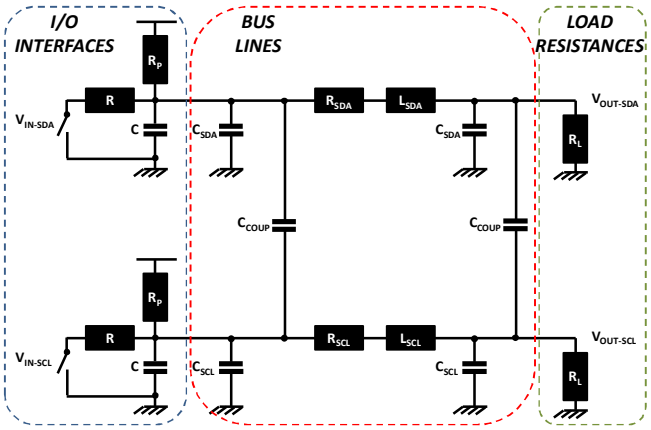


Fig.8. I²C bus simplified equivalent circuit.

Table I shows the selected configuration, which achieved suitable performance. At the end of the training process, equations (3) and (4) were obtained to approximate the relation between the inputs and outputs of the system, with *j* the hidden layer neuron number, *I* the input layer, O the output layer, and *d* the delay.

$$a_j = \text{tansig}\left(\begin{matrix}\sum_{d=1}^{D_I}(u(t-d)\times IW^{I,j}) \\ +\sum_{d=1}^{D_R}(y(t-d)\times LW^{O,j}) + b^j\end{matrix}\right) \quad (3)$$

$$y = purelin\left(\sum_{j=1}^{N}(a_j \times LW^{j,O})\right) + b^O \quad (4)$$

In the last stage, the neural network equations were implemented in SystemC-AMS, using both conventional and dynamic TDF. The time step for conventional TDF was set to 1 ns. For dynamic TDF, it was set to 3 ns when both SDA's and SCL's logic levels were stable and 1 ns when SDA and/or SCL were switching levels. The dynamic TDF configuration allowed a reduction of about 70 % of the amount of unnecessary computation, which sped up the simulation. Unnecessary computation typically occurred when the bus lines' logic levels were stable and no SI effect was induced.

Simulation results of this neural-network-based SystemC-AMS TDF modeling platform were compared with a reference model, the SystemC-AMS ELN platform introduced in [18]. Note that this platform was experimentally validated. Fig 9.a is a simulation of an all-functional model, which only shows the system ideal digital behavior. No SI effect can be detected. Fig.9.b and 9.c present the simulation of the SDA and SCL bus lines with the reference model and the SystemC-AMS TDF platforms. SI effects due to the line characteristics or crosstalk are clearly visible. We can also see that the TDF platforms accurately match the reference model's behavior.

Table II shows that the RAE (Relative Absolute Error) between the TDF models and the reference is inferior to 3.1 %. As for the simulation speed, the dynamic TDF model is noticeably faster than the reference. The overhead compared to an all-functional model platform is important, but the TDF non-functional modules provide a designer with more information to accurately analyze their application. Also, the simulation duration is still very reasonable (5-15 seconds).

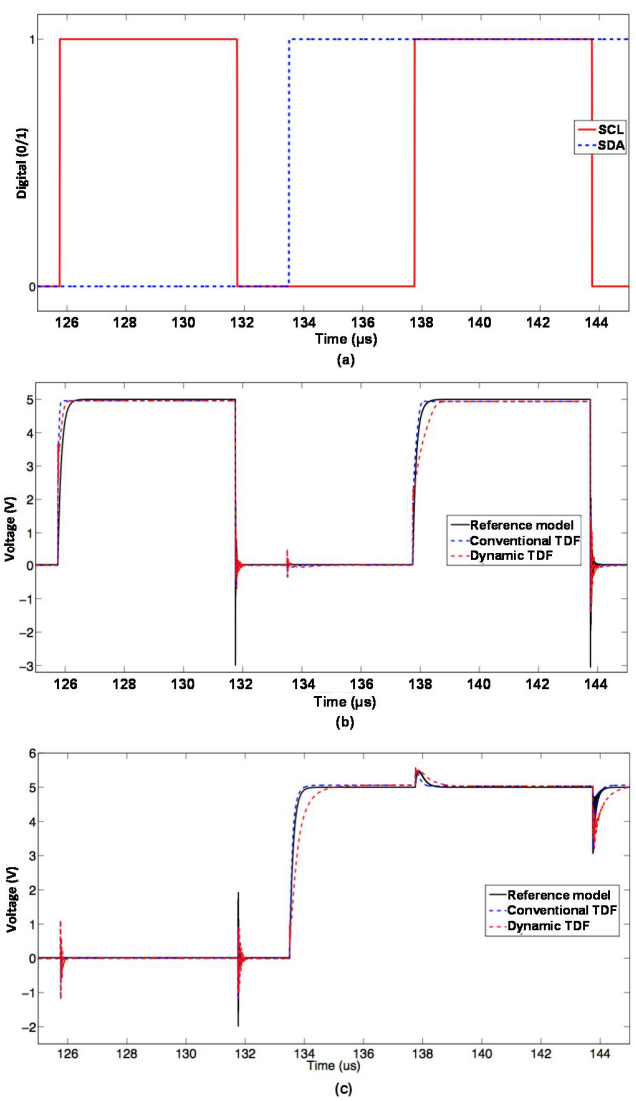


Fig.9. I²C platform simulation: (a) SCL and SDA lines, functional models only. (b) SCL line simulation with reference model, conventional and dynamic TDF. (c) SDA line simulation with reference model, conventional and dynamic TDF.

TABLE II
SIMULATION PERFORMANCES FOR I²C PLATFORM

	Line	*RAE*[1] *(%)*	*CPU Time*[2]
All-functional model	N/A	N/A	0.046 s
Reference model [18]	N/A	N/A	16.386 s
Conventional TDF model	SCL	2,91	13.190 s
	SDA	1,70	
Dynamic TDF model	SCL	3,10	5.246 s
	SDA	2,70	

[1] Relative Absolute Error
[2] On Intel Core i5-660, 3.33 GHz, RAM 2 Gbits

IV. USB 3.0 APPLICATION

The interest of the methodology now being showed on a simple case, a validation with a more complex application is proposed in this section. It also demonstrates that component models can be parameterized and combined in a modular way.

Fig.10. Illustration of the USB 3.0 application.

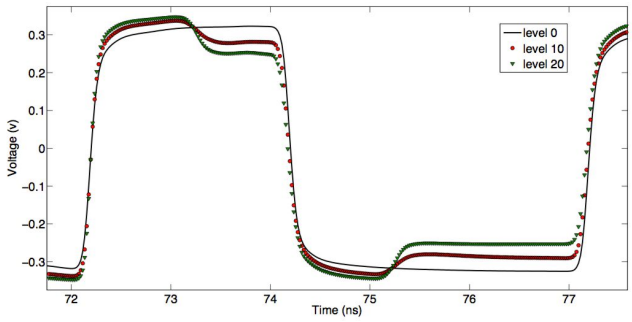


Fig.11. Influence of the 1st post-tap parameter.

A. *Platform presentation and reference model*

The method was applied to model the SI performances of an USB 3.0 transceiver application. In this section, the methodology is adapted to suit the requirements of a virtual prototyping library, that is models modularity, flexibility and simulation speed. To do so, platforms were built with one functional and one non-functional module per component (in the former I²C use case, the SI effects of the whole platform were modeled in a unique module).

The application is presented in Fig.10. The system featured two Altera Stratix IV FPGA [36], a transmitter (TX) and a receiver (RX), which exchanged data via an USB 3.0 link. Communication was performed by a High Speed Serial Interface (HSSI) module, which can be implemented in the FPGA device.

To improve communication, and reduce signal degradations due typically to reflection processes and unadapted transmission lines [37], HSSI modules can be parameterized, for instance to perform amplitude pre-emphasis in the emitter, or DFE equalization in the receiver. Fig.11 shows the influence of one of the pre-emphasis parameters (1st post-tap) on the emitted signal. Here, the input is weakly modified in amplitude to anticipate the signal modifications due to all SI effects.

To find a suitable configuration for the HSSI modules, Altera provides encrypted HSPICE models of both TX and RX modules. These models were used, in association with a S-parameter model of an USB 3.0 SuperSpeed cable, as a reference model for this application. Although this reference model is available and ready-to-use, its major drawback is its simulation speed: it takes approximately 45 minutes to simulate the transmission of 200 bits, at a 1 Gbps rate. Since there are 8192 possible configurations for the sole TX module, the use of these models is not convenient. It is also unrealistic to use them in a system-level simulation of a global system.

TABLE III
CONFIGURATION OF FTDNN NETWORK

Component	D_I	N	*epoch*	*Training duration*
TX	100	10	2000	29 min
Cable	200	10	4000	1 h 05 min
RX	50	10	2000	25 min

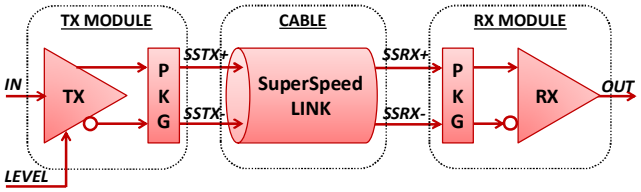


Fig. 12. Non-functional models of the USB 3.0 application.

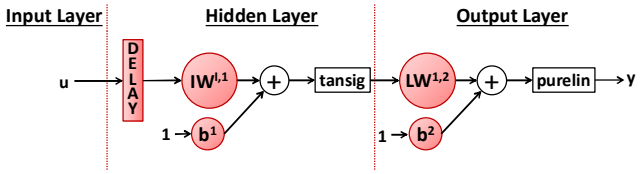


Fig.13. FTDNN network architecture.

B. *Non-functional modues design*

This application was modeled in a virtual prototyping library context, meaning that each component was designed as a separate block (in this case, a block features a functional model and a non-functional model). This introduces the modularity capabilities of our approach.

Fig.12 presents the non-functional models of the USB 3.0 application, based on the methodology. Since the focus of this paper is on the modeling of SI effects, the functional part of the platform (i.e. the tasks implemented in both FPGA) is not presented. The non-functional part of the model included three modules: TX, RX and cable. TX and RX took into account the influence of the FPGA package (PKG). Moreover, TX was parameterized by the LEVEL input, which set the transceiver's pre-emphasis level (the 1st post tap parameter). The same feature could have also been implemented in the RX module to take into account the DFE equalization and its configuration.

Each non-functional module was based on a specific neural network. The same architecture was used for all networks: Focused Time-Delay Neural Network (FTDNN) [38] (Fig.13). Indeed, FTDNN is less complex than NARX network. As a result, it requires less memory and time for training. However, it is possible to combine various architectures.

The FTDNN networks also had three layers: input, hidden, and output, but did not include a recurrent branch. Parameters to be set were the number of delay for input (D_I), the number of neurons in the hidden layer (N) and the number of iterations (*epoch*). Table III presents the chosen configuration for each module. At the end of the training process, equations (5) and (6) were obtained.

$$a_j = tansig\left(\sum_{d=1}^{D_I}(u(t-d) \times IW^{I,j}) + b^j\right) \quad (5)$$

$$y = purelin\left(\sum_{j=1}^{N}(a_j \times LW^{j,O}\right) + b^O \quad (6)$$

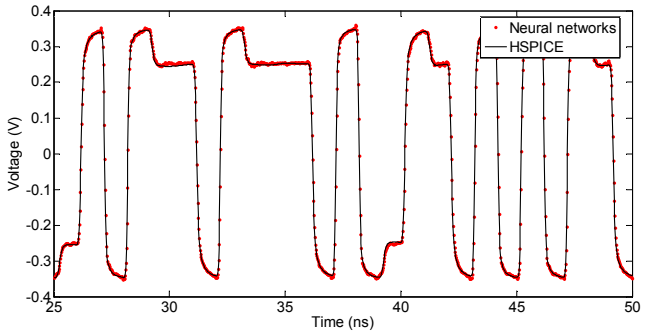


Fig.14. Transient behavior of OUT signal (HSPICE and SystemC-AMS simulation comparison).

TABLE IV
SIMULATION PERFORMANCES FOR USB APPLICATION

Model	*RAE (%)*	*CPU Time*	*Speedup*
HSPICE	N/A	2808.35 s	N/A
SystemC-AMS (case 1)	3.6	4.96 s	×567
SystemC-AMS (case 2)	1.5	1.47 s	×1910

Finally, these two equations were implemented in a dedicated SystemC-AMS TDF module for each component.

C. *Simulation results*

Simulation of the SystemC-AMS platform was compared to the HSPICE reference model. For this example, the input signal was a pseudo random bit sequence generated at the rate of 1 Gbps. Simulations were performed with different 1st post-tap levels (0, 10 and 20)

Fig.14 compares the SystemC-AMS simulation results with the reference model for a 1st post-tap level of 20. The association of the three non-functional modules achieved excellent accuracy. RAE between these modules and the reference model was around 3.6 % (cf. Table IV, case 1).

The SystemC-AMS platform also achieved a significant speedup (×567). Indeed, during the time required by the methodology to simulate the 8192 transmitter configurations, one could simulate less than 15 configurations with the HSPICE models. This speedup allows an efficient system simulation, combining functional and non-functional models. Note that if a unique block to model the three components had been used, as in section 3.D, speedup and REA would be improved (see Table IV, case 2). However, the well appreciable modularity would be lost.

With this approach, one could then imagine a virtual prototyping library, where each component would be represented by an ideal functional model, and a non-functional performance model, that could help validate the SI behavior of the system or detect potential problems. For instance, if one wanted to model a system that transmits data at a 10 Gbps rate (i.e. beyond the USB 3.0 requirements), an all-functional ideal simulation (cf. Fig.15.a) would not reveal any particular problem, whereas a non-functional simulation (Fig.15.b) would clearly show the incoherent signal behavior due to the inappropriate transmission rate.

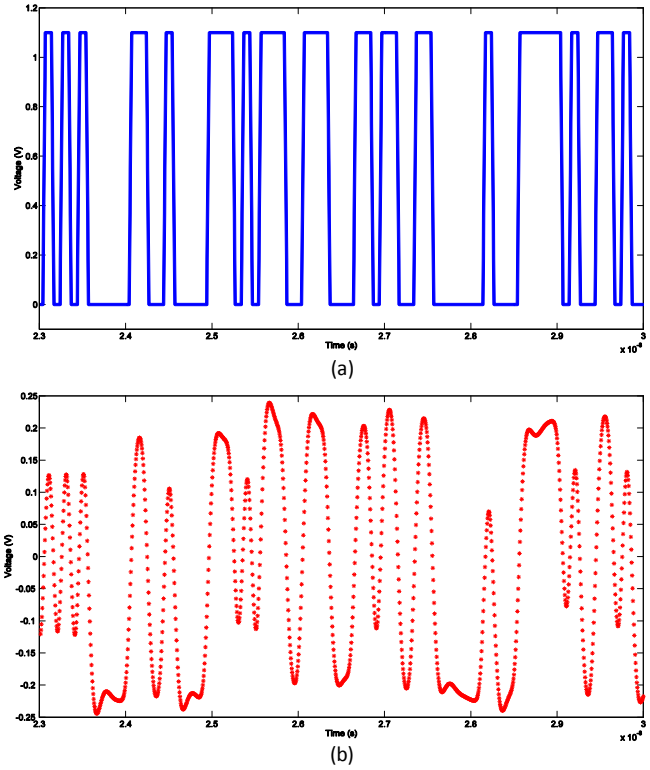


Fig.15. Simulation of a 10 Gbps rate data transmission. (a) All-functional simulation: no SI problem detected. (b) Functional and non-functional simulation. SI effects are detected.

The output signal eye diagram was also built (cf. Fig.16). Table V shows the eye height and width comparison between the SystemC-AMS models and the HSPICE reference. The eye opening characteristics were similar. Some of the differences between both diagrams were a consequence of the neural network approximation technique: data computed by the neural network concentrated around a few values, whereas the HSPICE simulation results were much more dispersive [38].

Indeed, the training of a neural network aims at finding the best parameters of a mathematical approximation function, in order to minimize errors. Therefore, it can't model all the situations but only some specific ones (the maximum, minimum, or interval boundaries) which are of interest to the designers.

V. CONCLUSION

In this paper, a novel methodology to model signal integrity at a high-level of abstraction for virtual prototyping of complex systems was introduced. Systems are modeled as a combination of functional modules, which give an ideal behavior of the application, and non-functional modules, which give the SI performances. These non-functional blocks are based on neural networks and are implemented in SystemC-AMS, using the TDF model of computation. The methodology achieves very good accuracy, while allowing significant simulation speedup, especially compared to SPICE-based models

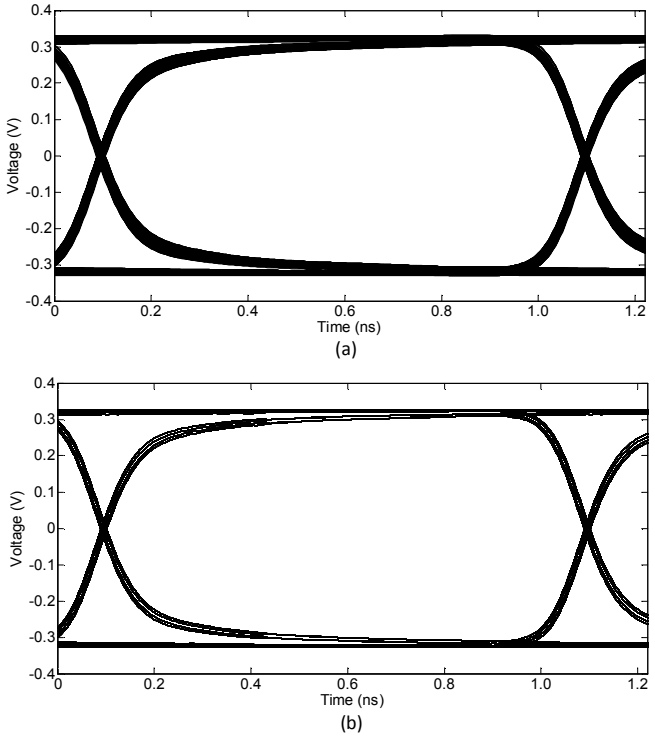


Fig.16. OUT node eye diagram (a) HSPICE simulation (b) SystemC-AMS simulation.

TABLE V
EYE DIAGRAM COMPARISON WITH HSPICE SIMULATION

Model	*Eye height difference (%)*	*Eye width difference (%)*
SystemC-AMS	2.1	1.0

For future works, additional blocks could be added. In the case of the USB application, DFE equalization or post-processing blocks could be implemented into the modeling platform, to perform for instance frequency domain conversion or stat eye analysis.

Virtual prototyping plays a major role in the design of electronic systems, since it allows fast simulation of the application behavior, with the help of a library of component models. Thanks to its modularity and parameterizability, the presented methodology could be used to enhance virtual prototyping tools, by adding early SI analysis capabilities. This would help designers in their tasks and improve their time-to-market.

REFERENCES

[1] M. Chiodo, D. Engels, P. Giusto, H. Hsieh, A. Jurecska, L. Lavagno, K. Suzuki, A. Sangiovanni-Vincentelli, "A case study in computer-aided co-design of embedded controllers", in Design Automation for Embedded Systems, vol. 1, no. 1, pp. 51-67, Jan. 1996, Kluwer Academic Publishers. DOI : 10.1007/BF00134683.

[2] J. Liu, X. Liu, E. Lee, "Modeling Distributed Hybrid Systems in Ptolemy II", in proc. of the American Control Conference (ACC), pp. 4984-4985, June 2001, Arlington, VA, USA. DOI:10.1109/ACC.2001.945773.

[3] K. Agarwal, D. Sylvester, D. Blaauw, "Modeling and analysis of crosstalk noise in coupled RLC interconnects", in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 25, no. 5, pp. 892-901, May 2006. ISSN: 0278-0070. DOI: 10.1109/TCAD.2005.855961.

[4] P. V. Hunagund, A. B. Kalpana, "Crosstalk Noise Modeling for RC and RLC interconnects in Deep Submicron VLSI Circuits", in Journal of Computing, vol. 2, no. 4, pp. 60-65, April 2010. ISSN 2151-9617

[5] J. Loeckx, G. Gielen, "Generic and Accurate Whitebox Behavioral Model for Fast Simulation of Analog Effects in Nanometer CMOS Digital Logic Circuits", in IEEE Transactions on Electromagnetic Compatibility, vol. 51, no. 2, pp. 351-357, May 2009. ISSN: 0018-9375. DOI: 10.1109/TEMC.2009.2014072.

[6] Y. M. Lee, Y. Cao, T. H. Chen, J. M. Wang, C. C. P. Chen, "HiPRIME: hierarchical and passivity preserved interconnect macromodeling engine for RLKC power delivery", in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 24, no. 6, pp. 797-806, May 2005. ISSN: 0278-0070. DOI: 10.1109/TCAD.2005.847938.

[7] R. Achar, M. S. Nakhla, "Simulation of high-speed interconnects." In Proc. of the IEEE, vol. 89, no. 5, pp. 693-728, May 2001. ISSN: 0018-9219. DOI: 10.1109/5.929650.

[8] N. Marques, M. Kamon, L. M. Silveira J. White, "Generating compact, guaranteed passive reduced-order models of 3-D RLC interconnects", in IEEE Transactions on Advanced Packaging, vol. 27, no. 4, pp. 569-580, Nov. 2004. ISSN: 1521-3323. DOI: 10.1109/TADVP.2004.831867.

[9] H. Kim, Y. Eo, "High-Frequency-Measurement-Based Circuit Modeling and Power/Ground Integrity Evaluation of Integrated Circuit Packages", in IEEE Transactions on Advanced Packaging, vol. 31, no. 4, pp. 910-918, Nov. 2008. ISSN: 1521-3323. DOI: 10.1109/TADVP.2008.2005472.

[10] J. Chen, L. He, "Piecewise linear model for transmission line with capacitive loading and ramp input", in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 24, no. 6, pp. 928-937, June 2005. ISSN: 0278-0070. DOI: 10.1109/TCAD.2005.847895.

[11] T. K. Wang, S. T. Chen, C. W. Tsai, S. M. Wu, J. L. Drewniak, T. L. Wu, "Modeling Noise Coupling Between Package and PCB Power/Ground Planes With an Efficient 2-D FDTD/Lumped Element Method", in IEEE Transactions on Advanced Packaging, vol. 30, no. 4, pp. 864-871, Nov. 2007. ISSN: 1521-3323. DOI: 10.1109/TADVP.2007.901764.

[12] Y. Eo, S. Shin, W. R. Eisenstadt, J. Shim, "Generalized traveling-wave-based waveform approximation technique for the efficient signal integrity verification of multicoupled transmission line system", in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 21, no. 12, pp. 1489-1497, Dec. 2002. ISSN: 0278-0070. DOI: 10.1109/TCAD.2002.804381.

[13] I. S. Stievano, L. Rigazio, I. A. Maio, F. G. Canavero, "Behavioral Modeling of IC Core Power-Delivery Networks From Measured Data", in IEEE Transactions on Components, Packaging and Manufacturing Technology, vol. 1, no. 3, pp. 367-373, Mar. 2011. ISSN: 2156-3950. DOI: 10.1109/TCPMT.2010.2099970.

[14] T. R. Cunha, H. M. Teixeira, J. C. Pedro, I. S. Stievano, L. Rigazio, F. G. Canavero, R. Izzi, F. Vitale, A. Girardi, "Validation by Measurements of an IC Modeling Approach for SiP Applications", in IEEE Transactions on Components, Packaging and Manufacturing Technology, vol. 1, no. 8, pp. 1214-1225, Aug. 2011. ISSN: 2156-3950, DOI: 10.1109/TCPMT.2011.2158313.

[15] W. Dghais, H. M. Teixeira, T. R. Cunha, J. C. Pedro, "Novel Extraction of a Table-Based I–Q Behavioral Model for High-Speed Digital Buffers/Drivers", in IEEE Transactions on Components, Packaging and Manufacturing Technology, vol. 3, no. 3, pp. 500-507, March 2013. DOI: 10.1109/TCPMT.2012.2234212.

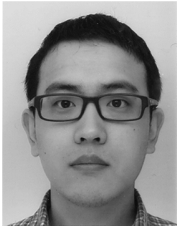
[16] K. Y. Yang, C. B. Chang, T. Y. Wu, W. S. Wang, Y. H. Lin, R. B. Wu, "Modeling and Fast Eye Diagram Estimation of Ringing Effects on Branch Line Structures", in IEEE Transactions on Components, Packaging and Manufacturing Technology, vol. 4, no. 4, pp. 641-647, April 2014. DOI: 10.1109/TCPMT.2013.2297320.

[17] M. Alassir, J. Denoulet, O. Romain, A. Suissa, P. Garda, "Modelling field bus communications in mixed-signal embedded systems"; in EURASIP Journal of Embedded Systems, vol. 2008, pp. 1-11, Jan. 2008 . DOI: 10.1155/2008/134798.

[18] M. Alassir, J. Denoulet, O. Romain, P. Garda, "Signal integrity-aware virtual prototyping of field bus-based embedded systems", in IEEE Transactions on Components, Packaging and Manufacturing Technology, vol. 3, no. 12, pp. 2081-2091, Dec. 2013. DOI: 10.1109/TCPMT.2013.2262151.

[19] A. Sangiovanni-Vincentelli, "Quo Vadis, SLD? Reasoning About the Trends and Challenges of System Level Design", in Proc. of the IEEE, vol. 95, no. 3, pp. 467-506, March 2007. DOI: 10.1109/JPROC.2006.890107.

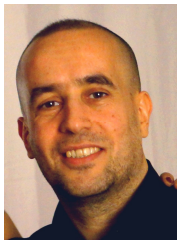
[20] IEEE Standard SystemC® Language, Reference Manual, 2005 http://standards.ieee.org/getieee/1666/download/1666-2005.pdf.
[21] M. Barnasconi, "SystemC AMS extensions: Solving the Need for Speed", DAC Knowledge center, May 2010; http://www.accellera.org/community/articles/amsspeed/community/articles/amsspeed/SystemC_AMS_Solving_the_Need_for_Speed_May_2010.pdf.
[22] Q. Zhang, K. Gupta, "Neural Networks for RF and Microwave Design", Norwood, MA: Artech House, 2000. ISBN: 1580531008.
[23] M. Hagan, H. Demuth, M. Beale, "Neural Network Design", PWS Publishing Company, 1996. ISBN: 0971732108.
[24] V. Devabhaktuni, M. Yagoub, Y. Fang, J. Xu, Q. Zhang, "Neural networks for microwave modeling: model development issues and nonlinear modeling techniques", in Int. Journal of RF and Microwave Computer-Aided Engineering, vol. 11, no. 1, pp. 4–21, Jan. 2001. DOI: 10.1002/1099-047X(200101)11:1<4::AID-MMCE2>3.0.CO;2-I.
[25] S. Haykin, "Neural Networks: A Comprehensive Foundation", Pearson Education, 1998. ISBN: 0132733501.
[26] L. Ljung, "System identification: theory for the user", Prentice Hall PTR, 1987. ISBN: 9780136566953.
[27] D. White, D. Sofge, "Handbook of Intelligent Control", New York: Van Nostrand Reinhold, 1992. ISBN: 0442308574.
[28] "Simulink-Dynamic System Simulation for MATLAB", Mathworks, 1996.
[29] D. Hush, "Classification with neural networks: a performance analysis", in IEEE International Conference on Systems Engineering, pp. 277–280, Aug. 1989. DOI: 10.1109/ICSYSE.1989.48672.
[30] M. F. Moller, "A scaled conjugate gradient algorithm for fast supervised learning", Neural Networks, vol. 6, no. 4, pp. 525-533, Nov. 1993. DOI: 10.1016/S0893-6080(05)80056-5.
[31] M. Barnasconi, K. Einwich, C. Grimm, A. Vachoux, T. Maehne, "Advancing the SystemC Analog/Mixed-Signal (AMS) Extensions: Introducing Dynamic Timed Data Flow", Open SystemC Initiative, Sept. 2011, http://accellera.org/images/resources/articles/amsdynamictdf/Whitepaper_SystemC_AMS_Dynamic_TDF_September_2011.pdf.
[32] R. Wang, J. Denoulet, S. Feruglio, F. Vallette, P. Garda, "High level modeling of signal integrity in field bus communication with SystemC-AMS", in Int. Conf. on Electron., Circuits and Syst.(ICECS), pp. 889–892, Dec. 2012. DOI: 10.1109/ICECS.2012.6463519.
[33] "I2C-bus specification and user manual." NXP Semiconductors. http://www.nxp.com/documents/user manual/UM10204.pdf.
[34] G. Vasilescu, "Electronic Noise and Interfering Signals: Principles and Applications", Chap. 12, Springer-Verlag, 2005. ISBN 3-540-40741-3.
[35] S. Chen, S. Billings, P. Grant, "Non-linear system identification using neural networks", in Int. Journal of Contr., vol. 51, no. 6, pp. 1191-1214, 1990. DOI: 10.1080/00207179008934126.
[36] "Stratix IV Device Handbook", vol. 2, Altera Corporation, 2012. https://www.altera.com/en_US/pdfs/literature/hb/stratix-iv/stx4_5v2.pdf
[37] "Understanding the Pre-Emphasis and Linear Equalization Features in Stratix IV GX Devices", Altera Corp., 2010. https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/an/an602.pdf.
[38] K. J. Lang, A. H. Waibel, G. E. Hinton, "A time-delay neural network architecture for isolated word recognition", Neural Networks, vol. 3, no. 1, pp. 23-43, June 1989. DOI: 10.1016/0893-6080(90)90044-L.
[39] R. Wang, J. Denoulet, S. Feruglio, F. Vallette, P. Garda, "Modeling of signal integrity in bus communications with timed data flow SystemC-AMS", in Forum on Specification Design Languages (FDL), pp. 1–6, Sept. 2013. ISSN: 1636-9874.



Ruomin Wang received the M.Sc. and Ph.D. degrees in electrical engineering from Pierre and Marie Curie University, Paris, France, in 2010 and 2014, respectively. His research interests include signal integrity, heterogeneous systems modeling, neural networks and microelectromechanical systems and additive manufacturing.

Julien Denoulet is an assistant professor at Pierre and Marie Curie University in Paris, France. He received an engineering degree from ESIEE in 2000, the MS degree in 2001, and the Ph.D degree in 2004 from University of Paris-Sud, Orsay, France, working on massively parallel architectures for image processing. He is now a member of the System on Chip department at LIP6 laboratory, in Pierre & Marie University, Paris, France. His current activities include modeling of heterogeneous systems and innovative architectures for System-on-chip.

Sylvain Feruglio (M'06) received the B.Sc. and M.Sc. degrees in electrical engineering and the master's degree in electronics, option instrumentation, and systems from the Université Pierre et Marie Curie (UPMC) - Paris 6 (France), in 1999 and 2001, respectively, and the Ph.D. degree in noise computation in integrated active pixel image sensors from LISIF, UPMC, in 2005. He then joined IMEP-LAHC, Institut National Polytechnique de Grenoble (France), working on the study and the characterization of new CMOS and SOI technologies. Since 2007, he has been with LIP6 in the System on Chip department. His research interests include integrated sensors and electronics, noise analysis and signal integrity, mainly applied to image sensors and biomedical engineering.



Farouk Vallette received the MS degree in 1990, and the Ph.D degree in 1993, from University Pierre and Marie Curie, France. Since 1996, he has been assistant professor at Pierre and Marie Curie University in Paris. After being part of the LISIF from 1996 to 2005, he has been member of the LIP6 in the System on Chip department since 2007. He worked especially on design and optimization technics using sensitivities computation.

Patrick Garda (SM'07) received the B.S. in Mathematics, M.Sc. in Computer Science and Ph.D. in Electrical Engineering degrees from Paris 11 University, Orsay, France, in 1978, 1980 and 1984 respectively. Between 1976 and 1985, he was with the ENS Cachan, France, first as student, then as Teaching Assistant. In 1985, he joined the IEF Laboratory, Orsay, France, as a CNRS Research Scientist, conducting research on artificial retinas, analog implementations of neural networks and parallel architectures. In 1995, he joined Pierre & Marie University, Paris, France, as Professor, conducting research on intelligent sensors and embedded architectures. He is currently member of the Electronic Systems group in the LIP6 Laboratory, Vice-Director of the ICT, mathematics, physics, and nanotechnologies department at the French Ministry of National Education, Higher Education and Research. His current research interests include the energy autonomy of connected devices and biomedical applications.