Virtual Prototyping of Pressure Driven Microfluidic Systems with SystemC-AMS Extensions

Víctor Fernández
University of Cantabria
Santander, Spain
victor@teisa.unican.es

Andrés Mena
AlphaSIP, S.L.
Madrid, Spain
software@alphasip.es

Cédric Ben Aoun,
François Pêcheux
UPMC, LIP6, Paris, France
{Cedric.Ben-Aoun,
francois.pecheux}@lip6.fr

Luis J. Fernández
University of Zaragoza
Zaragoza, Spain
luisf@unizar.es

***Abstract*— The design of "Lab on a Chip" microfluidic devices is, typically, preceded by a long and costly period of prototyping stages in which the system is gradually refined by an iterative process, involving the manufacturing of a physical prototype and the making of a lot of laboratory experiments. In this scenario, a virtual prototyping framework which allows the emulation of the behavior of the complete system is greatly welcome. This paper presents such a framework and details a virtual prototyping methodology able to soundly handle microfluidic behavior based on SystemC-AMS extensions. The use of these extensions will permit the communication of the developed microfluidic models with external digital or mixed signal devices. This allows the emulation of the whole Lab on a Chip system as it usually includes a digital control and a mixed-signal reading environment. Moreover, as SystemC-AMS is also being extended to cover other physical domains within the CATRENE CA701 project, interactions with these domains will be possible, for example, with electromechanical or optical parts, should they be part of the system. The presented extensions that can manage the modeling of a micro-fluidic system are detailed. Two approaches have been selected: to model the fluid analytically based on the Poiseuille flow theory and to model the fluid numerically following the SPH (Smoothed Particle Hydrodynamics) approach. Both modeling techniques are, by now, encapsulated under the TDF (Timed Data Flow) MoC (Model of Computation) of SystemC-AMS.**

Keywords—SystemC-AMS; Multi-Domain Simulation; Virtual Prototyping; Microfluidics; Lab on a Chip; Poiseuille Flow; SPH (Smoothed Particle Hydrodynamics).

I. INTRODUCTION

Cyber-physical systems usually encompass several physical domains in mutual interaction. The precise simulation of involved domains, by specific tools, is feasible (and sometimes done) but this is not usually profitable because it is usually very time consuming and the joint behavior of the system is not predicted. Moreover, it leads to a long iterative process which is, sometimes, replaced by a hardware prototyping process with real devices.

The opportunity for a unified simulation of different physical domains can be, up to date, partially addressed with existing research and commercial tools. On the research and open-source side, Ptolemy [1] and Modelica [2] have to be highlighted. The first one can have a wide variety of MoCs and, the second one, proposes a wide library of components (described with a specific object-oriented open language and supported by several free and commercial tools) in several physical domains. The presented approach works at a lower abstraction level than Ptolemy, enabling to give more physical details. Compared with Modelica, the presented approach allows an easier communication with digital control (and its embedded software) due to the integration into a SystemC environment.

On the commercial side, several tools are available and they usually rely on numerical methods like FEM (Finite Elements). Good examples of this kind of tools are COMSOL [3], ANSYS [4] or 3DS [5]. The use of these tools to describe a device implies long periods of time. Furthermore, they are highly computationally complex. They can be used for very detailed simulations of concrete locations but they are not suitable for fast prototyping of complete systems. On the other hand, they don't include the embedded software interaction. Other relevant tools on the commercial side are Matlab/Simulink/Simscape [6], LMS Imagine.Lab Amesim [7] and those based on previously mentioned Modelica language (Dymola [8], SimulationX [9], etc.). Again, the advantage of the proposed approach is the use of a standard language that will ease the seamless integration with digital, mix-signal and, in general, with other physical domains.

The advance in micro-nano-electronics fabrication allows the integration of multi-physical systems in single chips which are, in turn, highly integrated with microelectronic embedded systems. In this scenario, the goal of the CATRENE CA701 H-INCEPTION project [10] is to develop a design environment for multi-domain microelectronics assisted systems where the system definition, its design partitioning across the different physical domains, and its integral functionality can be analyzed and verified including the interaction with the overall application environment. On the modeling side, the language proposed is SystemC-MDVP where MDVP stands for Multi-Domain Virtual Prototypes.

SystemC-MDVP is itself an extension of SystemC-AMS [11] which is already an extension of SystemC [12]. The modeling of HW/SW parts as well as the AMS sections is well covered by well-known capabilities of SystemC-AMS. Other

physical domains have to be supported by defining and integrating the necessary MoCs.

One of the proof-of-concept multi-domain applications that will be modeled in the project is a prototype of a point-of-care blood analysis system which includes a microfluidic sub-system. The analysis procedure is controlled by a microcontroller which activates pistons and valves in order to move and mix the blood samples with several reagents that imply several biochemical processes. The final biochemical reaction is electrically monitored with an AMS device, digitally converted and sent back for characterization.

Micro and nano-fluidic devices are being widely used for several application areas: Clinical diagnostics, advanced sequencing, drug discovery, environmental monitoring and much more. There are several kind of devices considering the way they move and process the fluids but two main groups can be distinguished: flow based, with several mechanisms to manage the fluidic flows, and discrete droplet (also known as digital) based [13][14]. The microfluidic systems considered in the presented approach are pressure-driven flow-based.

Two modeling approaches have been followed for the emulation with SystemC-AMS extensions of pressure-driven microfluidic devices. The first one involves the analytical solution based on well-known Poiseuille flows [15][16]. This is explained with more detail in the section II of this paper. The second modeling scheme is based on the numerical and geometric solution called Smoothed Particle Hydrodynamics (SPH) [17][18]. This second approach is detailed in section III of this paper.

Both Poiseuille and SPH modeling approaches have in common the property of being simpler (and consequently faster) in the definition of the model and in the execution itself, than other more accurate numerical methods (FEM, for instance). The final goal is to be able to have an environment, based on SystemC-MDVP, for simulation of all the involved physical domains at a high-level of abstraction, which gives the user a rapid idea of the behavior of the whole system in order to reduce the number of prototyping stages. Furthermore, by simulation, some features can be checked that can be unaffordable by real prototyping, allowing the implementation of better products.

In Fig. 1, the SystemC-AMS extension approach, which is being developed and extended in the project, is depicted. The dark grey boxes represent areas of extension and the light grey boxes represent standard areas. For the modeling of new physical domains, it is sometimes necessary to add a new specific Model of Computation (MoC) and Solver. This was the case for the SPH approach because the current SytemC-AMS implementation does not directly support the modeling of physical systems following a Computational Fluid Dynamics approach. Such approach needs a specific geometric part to represent 2D or 3D geometry and a specific solver for the precise numerical approach. For the approach based on the analytical Poiseuille equations, a specific library of microfluidic components has been developed which uses the existing TDF and ELN MoCs from current version of SystemC-AMS.

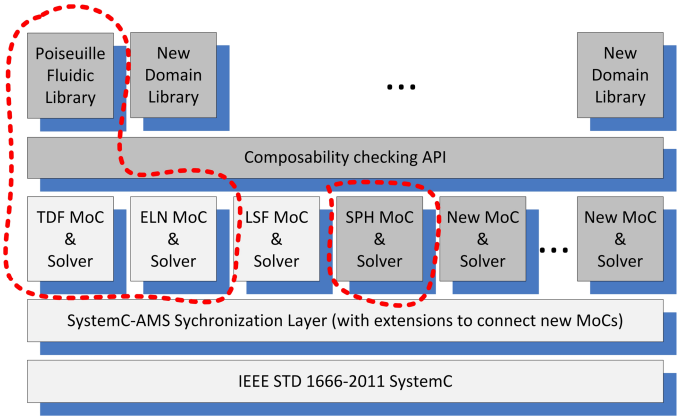

Fig. 1. SystemC-AMS extensions (dark grey). Locations of presented approaches (dashed line).

Experimental results will be illustrated in section IV and conclusions are given in section V.

A preliminary and immature description of the methodology applied and very incipient results of the presented work can be seen in the conference paper [19].

II. MODELING BASED ON HAGEN-POISEUILLE

The dynamic behavior of a fluid, which is supposed as continuum matter, is well represented by classical conservation laws. These are the conservation of mass, conservation of linear momentum (also known as Newton's Second Law of Motion), and conservation of energy (also known as First Law of Thermodynamics). The momentum axiom, applied to fluids, becomes the well-known Navier-Stokes equation. (1) Represents such equation supposing an incompressible flow with a constant viscosity.

Inertia (per volume) — Divergence of stress

$$\rho\left(\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v}\cdot\nabla\mathbf{v}\right) = -\nabla p + \eta\nabla^2\mathbf{v} + \mathbf{f} \quad (1)$$

Unsteady acceleration — Convective acceleration — Pressure gradient — Viscosity term — Other body forces

Where ρ is the density of the fluid, v is the flow velocity, p is the pressure and η is the viscosity. "Other body forces" are typically gravity, centrifugal or electromagnetic forces. When considering geometry, Navier-Stokes equations are nonlinear partial differential equations which cannot be solved analytically in the general case. That is due to the convective term which represents the variation of velocity with respect to the position. This phenomenon can be observed through a typical case: when a liquid flows into a tube that changes its cross-section. If the section becomes smaller, the speed has to be increased (with position, not with time) as the fluid is incompressible and in order to maintain a constant volumetric flow rate.

Under a set of approximations, that is, incompressible flow with a constant viscosity, steady laminar flow and long quasi-unidirectional transporting channels and viscous forces dominating other forces, the behavior of a fluid into a channel

can be modeled by equation of Hagen-Poiseuille (2). It describes the volumetric flow rate Q (in microliter/second for instance) as a linear variation with the gradient of pressure Δp. The constant of proportionality is named as hydraulic resistance R_h.

$$Q = \frac{\Delta p}{R_h} \tag{2}$$

The hydraulic resistance is a constant parameter which depends on the viscosity of the fluid and the geometry of the channel. For a circular shape the resistance has the value shown in (3), where η is the viscosity of the fluid, r is the radius of the pipe and L is the distance in which the gradient of pressure is considered.

$$R_{h(circle\_pipe)} = \frac{8\eta L}{\pi r^4} \tag{3}$$

For other profiles, square, rectangle, triangle, etc., the hydraulic resistance has a different equation [16]. The microchannels in current microfluidic systems are typically square (4) or rectangular like (5), where h and w are the height and width dimensions.

$$R_{h(square\_pipe)} = \frac{12\eta L}{1 - 0.917 \times 0.63}\frac{1}{h^4} \tag{4}$$

$$R_{h(rectangle\_pipe)} = \frac{12\eta L}{1 - 0.63\frac{h}{w}}\frac{1}{h^3 w} \tag{5}$$

Equation (2) is a simple linear equation that adapts quite well to microfluidic systems as they can be made with large numbers of microchannels and because those microchannels often have a geometry that leads to flows that are well approximated by the unidirectional solutions. On the other hand, (2) defines a behavior which is equivalent to an electrical circuit behavior under the Ohm's law with pressure acting as the voltage and the volumetric fluidic flow acting as a current. With this, Hagen-Poiseuille law and the conservation of mass axiom can be applied to compute pressures and volumetric flow rates in the microfluidic device using the same algebraic equations and solving methods that are used in electrical linear networks (which apply the Kirchhoff equations).

With the aim of boosting the modeling process by the end-user, a modeling strategy based on pre-defined fluidic components has been developed. Components defined are pipes, pumps, fluid providing tanks, chambers, valves, switches and waste points.

In the Poiseuille approach, each fluidic component is divided into two sub-components, an ELN (using the Electrical Linear Network MoC of SystemC-AMS) one and a TDF (using the Timed Data Flow Moc of SystemC-AMS) one.

The ELN MoC of SystemC-AMS is based on the instantiation of predefined linear network primitives such as resistors, current/voltage sources, capacitors, inductors, etc. which are used as macro models for describing the continuous-time relations between voltages and currents.

Due to the analogy of a fluidic network with an electrical one, the ELN MoC is used to compute the drop of pressure between the terminals of any component in the network.

The TDF MoC of SystemC-AMS is a discrete-time modeling style, which considers data as signal sampled in time. Each set of connected components defines a *time step*. Every *time step*, each module read its input samples, compute what is internally defined in a *processing()* function and the results are written to output ports.

The TDF part of each fluidic component has ports which pass a data of *class fluid*. The *class fluid* has three *double* variables: viscosity, volume and position. It also contains several functions for setting and getting previous values as well as functions for modifying the position and volume of the fluid as it passes through components. Moreover, *==* and *!=* operators have been overloaded to support operators of *fluid class* in order to compare different objects of such *class*.

Next, a description of the fluidic components will be shown. As an illustrative example, the pipe component is described with more detail.

A. Pipe

In Fig. 2, the structure of a pipe is shown. We have created a separate namespace for the Poiseuille models called PFN which stands for Poiseuille Fluidic Networks. As commented above, there are two sub-components: The ELN sub-component and the TDF sub-component.

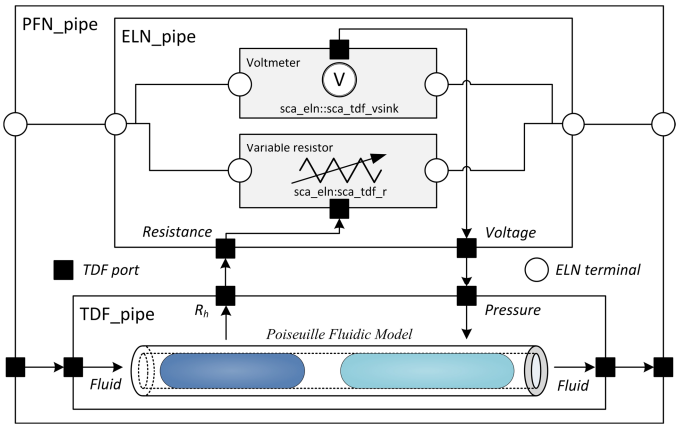

Fig. 2. Two SystemC-AMS components to describe one fluidic pipe

The ELN sub-component (ELN_pipe in Fig. 2) of the PFN component (PFN_pipe in Fig.2) has a variable resistor implemented with a *sca_eln::sca_tdf_r* ELN SystemC-AMS component. Its resistance value comes from the TDF part and it is connected to the ELN sub-components of the rest of the fluidic system. The value of voltage between the terminals of the resistor is captured by a voltmeter, implemented with a *sca_eln::sca_tdf_vsink* ELN SystemC-AMS component, and sent to the TDF part to be used as a pressure value.

The TDF sub-component (TDF_pipe in Fig. 2) computes, *time step* to *time step*, the values for fluidic positions inside the

pipe, taking into account the drop of pressure sent by the ELN sub-component, the type of fluids (actually, its viscosity) and the physical dimensions and shape of the pipe (applying the Poiseuille equations). With that information, a new value of resistance is computed and sent to the ELN part. For the example of Fig. 2 the value of the resistance will be the sum of the values due to the two liquids inside the tube (the air is assumed as another fluid with near-zero resistance). The TDF part is also connected to other TDF parts of other PFN components. They are connected through a TDF port which will pass a value of *class fluid*. This class will contain the physical properties of the fluid which is, at that moment, touching the port.

A pipe can contain several fluids. To do that, the fluids are stored in a *std::list<fluid>* data. Every fluid object in the pipe knows its position. Using the drop of pressure value (sent by the ELN sub-component) and the total resistance value of the pipe, the flow of liquid can be computed with the Poiseuille equation. With that value, the speed of fluids inside the pipe can be easily calculated. Finally, considering the dimensions of the pipe, the *time step* value and the volume of each fluid, the positions are updated. When a fluid is passing from one component to another, the volume is reduced accordingly in the first component and increased in the second one. The *time step* value has kept low in order to avoid relevant inaccuracies.

B. Pump

Fig. 3 shows the structure of a constant pressure pump. The model is composed of a voltage source, modeled with a *sca_eln::sca_tdf::sca_vsource* SystemC-AMS ELN primitive, that generates the constant voltage (pressure). The value of the generated voltage (pressure) is determined by a primary input (*pump_control* in Fig. 3) from user or environment. The model also includes an internal pipe because it models usual pumps behavior, like voltage sources usually have a serial resistance.

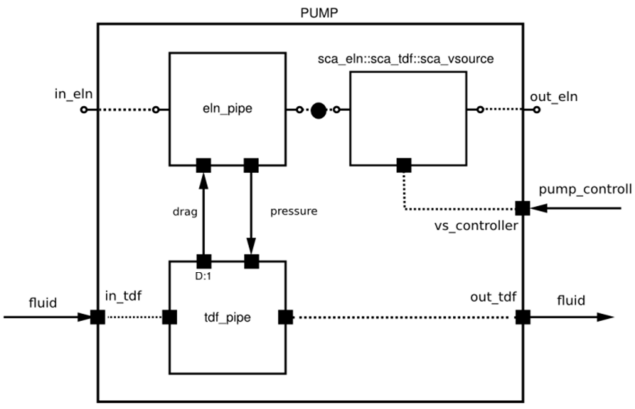

Fig. 3. PFN constant pressure pump

A constant flow pump has basically the same structure but with a current source instead of a voltage source, modeled with a *sca_eln::sca_tdf::sca_isource* SystemC-AMS ELN primitive.

C. Fluid providing tank

Its purpose is to serve as a limited source of the fluids that will be provided to the fluidic network. It is composed, as described in Fig. 4., by three main elements: *tdf_tank*, *eln_pipe* and a voltage source *sca_vsource*. The aim of the voltage source is to include a pressure to inject the fluid. The model of the *tdf_tank* is equivalent to a pipe but due to the big volume it sends a very small resistance value to the electrical part.

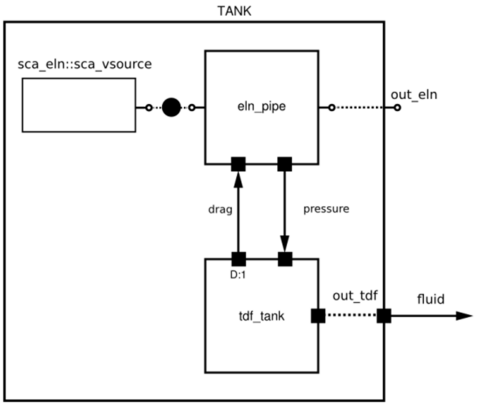

Fig. 4. PFN fluid providing tank

D. Chamber

A chamber is a storage component in the middle of the fluidic network used to store temporarily fluids or to perform special operations with them, like incubation, biochemical reactions, magnetic trapping of some components, measurements, etc. Since all these special operations are not implemented yet, chambers are used now only as storage.

The implementation and behavior of a chamber are very close to the pipe description except that a chamber has a bigger radius which gives it enough volume to store fluids and much less fluidic resistance.

E. Valve

The aim of this device is to open or close the flow of the liquid. This is simulated by opening or closing the equivalent electric circuit, with the *rswitch* primitive of the ELN MoC. There is a primary input *switch* to open or close the flow. The current version doesn't include an internal resistance as it emulates the behavior of a valve that does not interfere with the rest of components when it is open. If this is not the case, an internal or external pipe can be easily included. Fig. 5 shows the structure of a valve.

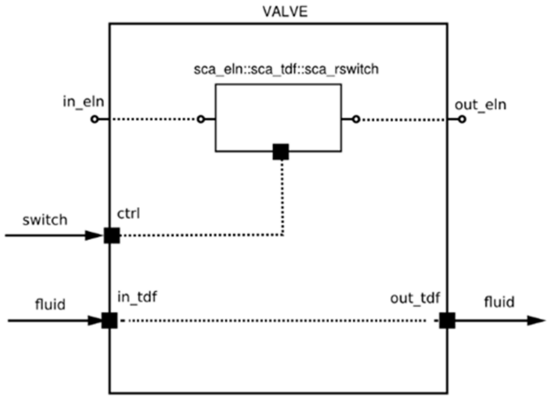

Fig. 5. PFN valve

F. Switch

The task of a fluidic switch device is to change the route of the fluidic flow. Two versions are implemented: the *in_switch* (Fig. 6) with two inputs and an output and the *out_switch* (Fig. 7) with one input and two outputs. Internally, they are implemented with two complementary valves controlled by the same primary input *switch_ctrl*. When one of the valves is open the other one is closed, and vice versa.

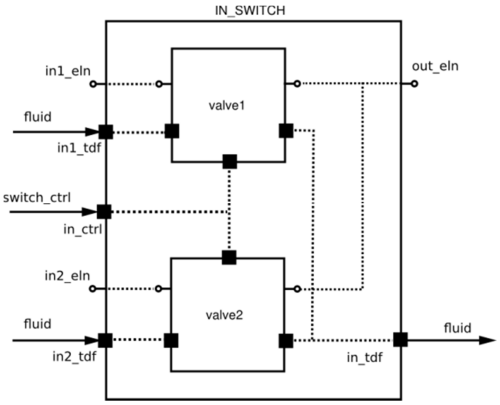

Fig. 6. PFN in_switch

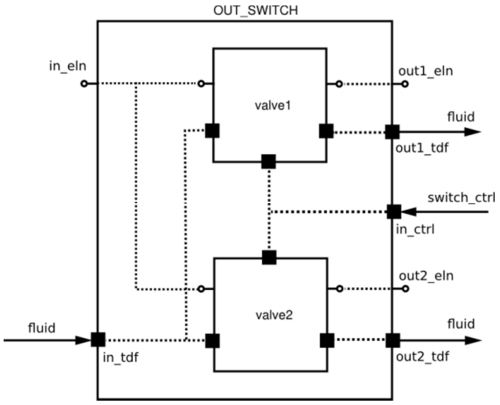

Fig. 7. PFN out_switch

G. Waste

This device represents a sink point in the network. The ELN part it typically connected to an electrical ground reference. In that case, the pressures applied have to substrate the atmospheric pressure not considered in the waste point. As an example, in the results section, a fluidic system which is composed by a providing tank connected to a fluidic path that ends in a waste point will be shown. The pressure value used in the tank is defined by the difference of pressure between the beginning and the end of the fluidic path due to the height of the fluid in the tank.

The TDF part of the waste component simply collects the incoming fluids and removes them from the flow.

H. Composition of a fluidic network

As it can be seen in the previous descriptions, the PFN components have three kinds of ports.

- Primary inputs for controlling the values of pressures and flows applied. Also, inputs for controlling the flow paths via valves and switches. These inputs are defined in the TDF Model of Computation. Therefore, when the system is simulated, TDF inputs have to be injected.
- TDF inputs and outputs for emulation of fluidic flow applying Poiseuille equations.
- ELN connections in order to emulate the equivalent electrical circuit. The goal is to compute the drop of pressure for any component.

In order to facilitate the connection's task, and also to reduce connecting errors, a mechanism based on the overloading of the >> operator has been implemented.

Following a classical SystemC-AMS style, the code to connect a simple case of one switch with three pipes is show in Fig. 8.

```
// Binding elements
sca_eln::sca_node eln_node1, eln_node2, eln_node3;
sca_tdf::sca_signal<fluid> tdf_signal1, tdf_signal2,
tdf_signal3;

// Connections
pipe1.out_eln->bind(eln_node1);
switch.in_eln->bind(eln_node1);

pipe1.out_tdf->bind(tdf_signal1);
switch.in_tdf->bind(tdf_signal1);

pipe2.out_eln  ->bind(eln_node2);
switch.in_eln_2->bind(eln_node2);

pipe2.out_tdf  ->bind(tdf_signal2);
switch.in_tdf_2->bind(tdf_signal2);

switch.out_eln->bind(eln_node3);
pipe3.in_eln  ->bind(eln_node3);

switch.out_tdf->bind(tdf_signal3);
pipe3.in_tdf  ->bind(tdf_signal3);
```

Fig. 8. Connection of PFN components with SystemC-AMS style.

The complexity of this code grows very fast with the number of devices to be connected. This is very time consuming and prone to errors. With the implemented operator overloading, the binding step is simplified. The code associated to the description of the same example, can be seen in Fig. 9.

```
*pipe1 >> *in_switch >> *pipe3;
*pipe2 >> *in_switch;
```

Fig. 9. Connection of PFN components with overloaded >> operator

III. SPH MODELING

Another way for emulating the fluidic behavior can be achieved using the Smoothed Particle Hydrodynamics (SPH) theory. The goal is the same as with the approach presented in previous section (modeling based on Hagen-Poiseuille equations), that is, to use a methodology, based on the SystemC-AMS standard language, which allows a user to compose and simulate a microfluidic system without paying the price for a thorough finite elements description and simulation. Both proposed modeling methodologies will be compared with a typical FEM approach and with experimental data in the results section of this paper.

In the SPH theory, the continuous fluid is replaced by a set of particles whose individual motion is approximated, and which, therefore, possess individual properties such as density, pressure, velocity, etc. The particles move according to the governing conservation equations, i.e. a simplified version of the well-known Navier-Stokes (1) equation, in which the convective acceleration term is not considered.

The right hand side of (1) represents the forces applied on a specific fluid particle. These forces can be divided into two categories: Internal Forces (such as Pressure, Viscosity, Surface Tension) and External Forces (Gravity, Magnetic Fields, etc.). All the forces involved in the SPH algorithm are expressed as density forces.

Any SPH quantity (density, force) for a given particle *i* can be determined using (6).

$$A_i(\vec{r_i}) = \sum_j A_j \frac{m_j}{\rho_j} W(\vec{r_i} - \vec{r_j}, h) \quad (6)$$

In (6) h represents the support radius (the distance of interaction of a particle with others) and W represents the Smoothing Kernel, which is used to weight the approximated implication of a particle j in the calculation of a quantity for particle i according to their Euclidean distance. r represents the position of a particle. See Fig. 10.

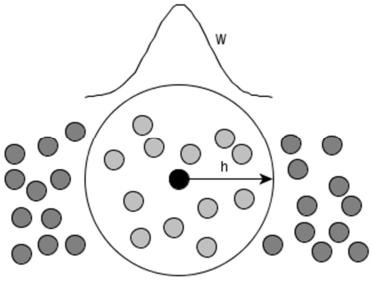

Fig. 10. The SPH support radius and smoothing kernel.

Applying (6) to the density computation leads to (7). The density of particle *i* represents the density of the area covered by the smoothing length.

$$\rho_i(\vec{r_i}) = \sum_j m_j W(\vec{r_i} - \vec{r_j}, h) \quad (7)$$

The generic equation (6) is also used to compute forces, but it has to be slightly modified. For instance, the modification needed to compute the pressure force while respecting the reciprocity principle of Newton Law implies (8).

$$\overrightarrow{F_{p_i}}(\vec{r_i}) = -\rho_i \sum_{j \neq i} m_j \left(\frac{P_i}{{\rho_i}^2} + \frac{P_j}{{\rho_j}^2}\right) \nabla W(\vec{r_i} - \vec{r_j}, h) \quad (8)$$

To compute the pressure P, a modification of the perfect gas equation "$P_i = k(\rho_i - \rho_0)$" is used, where ρ_0 represents the rest density of the fluid described by the set of particles. The integration of the rest density allows the representation of repulsion and attraction processes. A low density leads to a negative pressure, which induces on neighboring particles an attractive force. On the contrary, a high density causes a positive pressure leading to a repulsive force.

Along with the viscosity force, we take into account the relative velocity between particle *i* and its neighbors in (9), where η represents the viscosity coefficient.

$$\overrightarrow{F_{V_i}}(\vec{r_i}) = \sum_{j \neq i} \eta \frac{m_j}{\rho_j} (\vec{v_j} - \vec{v_i}) \nabla^2 W(\vec{r_i} - \vec{r_j}, h) \quad (9)$$

The force associated to the surface tension needs to meet some requirements. This force acts at the surface of the fluid and at the interface between two fluids, e.g. the interface between water and air. Since this force makes sense only at the surface, only particles which limit this surface should be affected. Several equations are used to achieve this purpose: (10), (11) and (12).

$$\overrightarrow{C_{s_i}}(\vec{r_i}) = \sum_{j \neq i} \frac{m_j}{\rho_j} W(\vec{r_i} - \vec{r_j}, h) \quad (10)$$

$$\vec{n_i} = \nabla C_{s_i} \quad \text{and} \quad \vec{K_i} = -\frac{\nabla^2 C_{s_i}}{|\vec{n_i}|} \quad (11)$$

$$\overrightarrow{F_{s_i}}(\vec{r_i}) = \sigma \vec{K_i} \vec{n_i} = \sigma(\nabla^2 C_{s_i}) \frac{\vec{n_i}}{|\vec{n_i}|} \quad (12)$$

C_s identifies the particles at the surface of the fluid. The gradient of this quantity for a particle allows us to determine if it should be affected by the force. If the length of this gradient is greater than some threshold (defined according to the fluid simulated) the particle should be affected by the surface tension.

The last force implied in the computation is the gravity, which is simply defined as $\overrightarrow{F_{g_i}} = \rho_i \vec{g}$.

With all the forces defined, we can compute the acceleration, in a straightforward way as $\vec{a_i} = \sum \vec{F_i} / \rho_i$, As we are using density forces, the division is performed on density and not mass.

As was mentioned in the introduction, a specific solver has been implemented. With this solver, the SPH simulation is performed in four steps. The first step does the computation of density (other quantities rely on the density value of each particle), the second one calculates the internal and external forces applied to each particle and the third one uses Newton's law to determine the acceleration. Finally, with an Euler, Leapfrog or Verlet scheme [18], acceleration is integrated twice to give the new position of each particle. The simulation loop can be seen in Fig. 11.

```
Simulation loop:
        Update density and pressure;
        Update forces;
        Update acceleration;
        Update position and velocity;
End loop
```

Fig. 11. SPH Simulation Loop

To describe any micro-fluidic network appropriately, the SPH algorithm must also consider the geometrical aspects. Every particle has to interact with the environment (pipe, tank, etc.). Therefore, we need to introduce an algorithm to detect when a particle is colliding with its container. In the current implementation, we used an "a posteriori" detection mechanism, which means that the collision is actually detected after it occurred. In the previously described simulation loop (Fig. 11), after updating the particle's position, we check against the containers if a collision occurred and if so the particle is moved and a correction on its velocity is applied according to the angle of the collision and the restitution coefficient of the container.

To reduce the overall computational cost some optimizations can be done. As all the quantities related to a particle *i* depend on the quantities of the neighbour particles, a dynamically managed grid can be used to contain sets of particles (all particles are stored into cells whose width is twice the smoothing length). Thanks to this structure, the search for neighboring particles consists in only looking at the cell adjacent to the cell which contains the particle. Second, referring to the third law of Newton, one can directly take into account the implication of particle *i* to particle *j* when we evaluate implication of particle *j* to particle *i* (reciprocity principle). This optimization is considered in the simulation loop, the first step before computing the density is to distribute the particle inside the grid according to its specific position.

In essence, the SPH model of computation that is to be integrated into SystemC MDVP will essentially mimic the primitive principles of the LSF (Linear Signal Flow) MoC in SystemC-AMS. The aim of this approach is, like in the Hagen-Poiseuille based approach, to provide predefined fluidic components to the end-user. The current primitives available are depicted in Fig. 12.

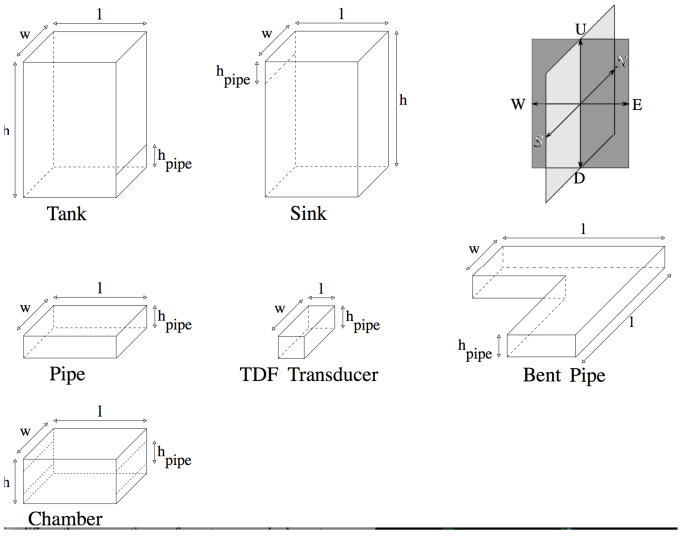

Fig. 12. The SPH modeling primitives.

To each geometric primitive corresponds a C++ constructor method that takes three sets of parameters. The first set of parameters represents the geometric bounding box of the SPH component (expressed as length, width and height). The second set of parameters defines the amount of fluid that is already present in the geometric element when the simulation starts. Finally, a last parameter allows modifying the orientation of a specific output port of the primitive, opening the way to the building of any 3D topology of the fluidic network. This solution greatly simplifies the connections of upstream and downstream elements and the global design. The composition scheme of SPH primitives relies on the standard SystemC TLM binding procedure, i.e. the SPH target element port in is bound to the corresponding SPH master element port out.

Fig. 13 shows a simple fluidic netlist composed of a tank, a pipe, two bent pipes instances and a sink.

```
#include <sc_mdvp.h>
[…]
#include <sph.h>
int main(){
  […]
scm_sph::sph_tank t1(l=3000, w=3000, h=5000, hpipe=1000, Fluid(),
OUT_EAST) ;
scm_sph::sph_pipe p1(l=10000, w=3000, h=1000, NULL) ;
scm_sph::sph_bentpipe bp1(l=2000, w=3000, h=1000, NULL,
OUT_SOUTH) ;
scm_sph::sph_bentpipe bp2(l=2000, w=3000, h=1000, NULL,
OUT_EAST) ;
scm_sph::sph_sink s1(l=3000, w=3000, h=5000, hpipe=1000, NULL) ;

t1.out(p1.in) ;
p1.out(bp1.in) ;
bp1.out(bp2.in) ;
bp2.out(s1.in) ;

  […]

}
```

Fig. 13. Composition of SPH primitives

After the SPH netlist has been elaborated, the simulator elaboration phase converts the individual geometric volumes associated to each SPH primitive into a global complex 3D volume that will act as the SPH simulation envelope. All the SPH simulation process is performed in this 3D envelope.

SPH is intrinsically very sensitive to simulation parameters. Therefore, a strong process of tuning was carried out in order to match the simulation results with real experiments.

3D visualization is obtained by means of a classic C++ OpenGL rendering engine which represents fluidic primitives either as parallelepipedic structures (Cartesian coordinates) or spheres and cylinders (Spherical coordinates). The latter simplifies computation as collisions between particles and the fluidic primitives involve fewer calculations. The camera and lookat 3D points can be modified by the end-user to identify in the 3D fluidic network a region of interest and to be able to zoom on it.

The SPH-TDF transducer can be used to detect the presence of specific particles in a given volume. These particles are referenced, and a TDF-compatible scalar value is generated according to the ratio of counted particles over the volume.

IV. RESULTS

In order to evaluate the proposed modeling schemes, a real prototype is tested and its timing behavior is compared with the PFN and SPH simulation results. Moreover, such simulation data is also compared with those provided by a typical FEM simulation framework.

The testing device was designed to include 5 different ports (Fig. 14) which can be used as inlets or outlets, and 2 microchambers. This configuration is frequently used for diagnostic applications, using the first microchamber for sample concentration, mixing and purification. The second chamber usually includes electrodes or microsensors to finally detect the molecule of interest.

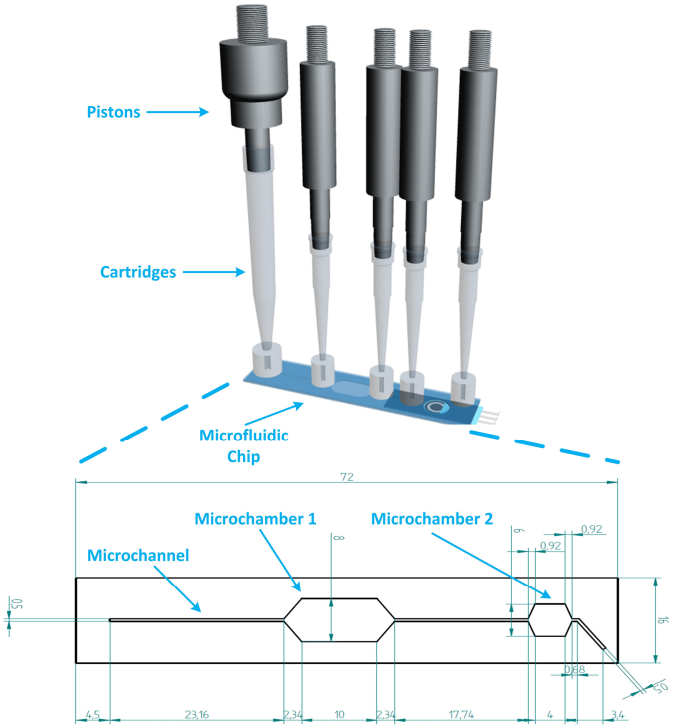

Fig. 14. Microfluidic chip for testing

The microfluidic device was made of COP by lamination techniques. First, two 188 µm thick sheets were first structured by a cutting-blade to obtain the desired configuration of microchannels and microchambers. Another two COP layers were then structured, one to be used as bottom layer and the other with included holes at the port locations to be used as inlets and outlets. Layers were aligned and bonded together using pressure sensitive adhesives (PSA). As a result, the microchannels and microchambers presented in figure 14 were created with an approximate depth of 375 µm.

In order to get timing data, a fluid was injected from left end port and it was collected in right end port. This test was made in different conditions that will be exposed below. During the fluidic path from left to right, 6 positions are distinguished as depicted in Fig. 15. The time the fluid last to get to each point was measured.

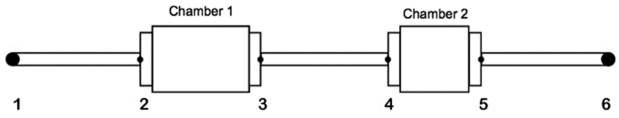

Fig. 15. Time measuring points in the microfluidic device

As an illustration of the simulation models used, in Fig. 16, a representation of the PFN model is exposed. The figure shows a constant voltage (pressure) pump model as injecting device. This model was used for all the tests except for the case of constant injection rate. In that case, a constant current (flow) pump model was used.

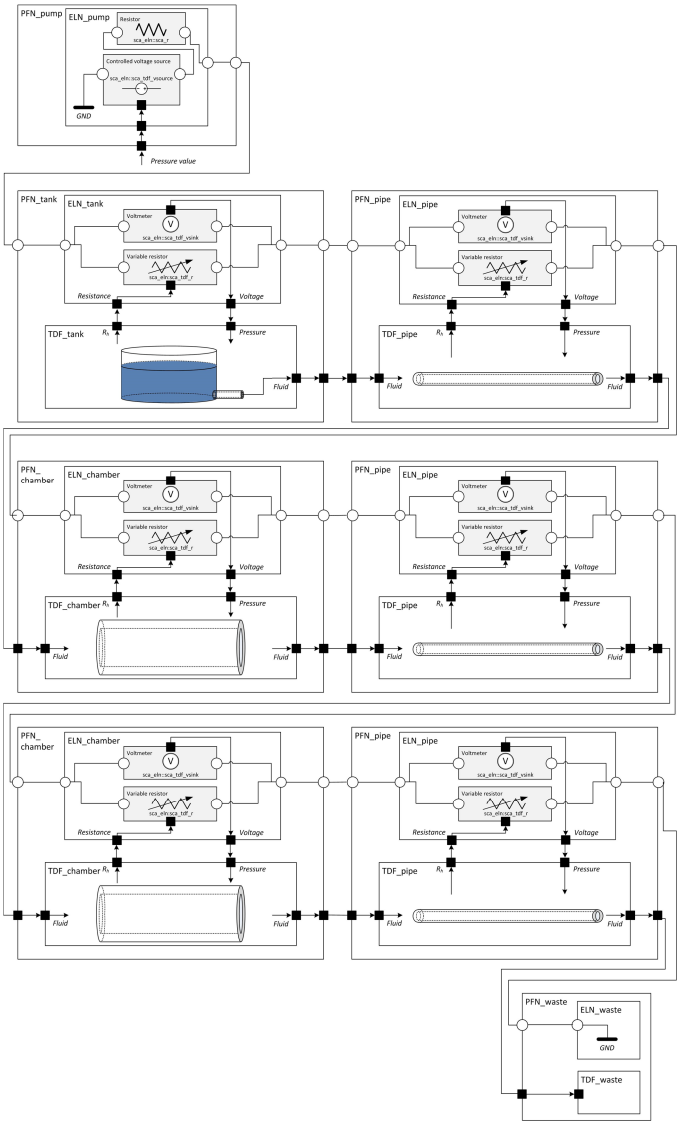

Fig. 16. PFN model of the tested microfluidic chip

Figure 17 presents the SPH representation of the fluidic network of Figure 15, at various simulation stages. A complementary tank has been connected to the input point 1 in order to modulate the pressure on the particles at this location, to match the experiments and the PFN model.

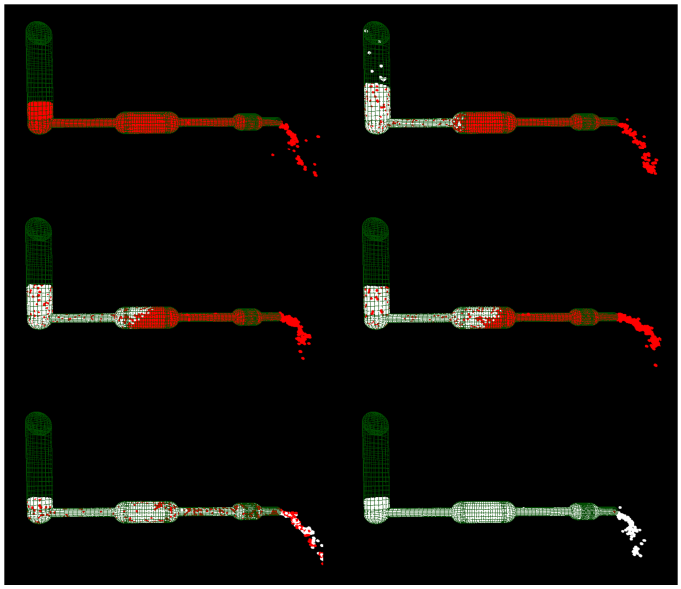

Fig. 17. SPH model of the tested microfluidic chip

Next, different experiments and comparison results are broken down.

A. Constant rate

A constant liquid flow was imposed in the microfluidic device pushing the plunger of the syringe placed at the inlet at a constant speed. The speed used and the diameter of the syringe resulted on a flow rate of 0,014 ml/min, and the liquid used was water. To visualize the speed at which the liquid is introduced within the microchannels and microchambers, a mixture of Rhodamine B and water was inserted in the syringe. First, the microdevice was completely filled by water, then the syringe was activated, and the advance of the characteristic blue colour of Rhodamine B was optically observed by a microscope. The times at which the mixture reached the different fluidic ports are shown in Fig. 18, and they are compared with the results obtained by the simulation.

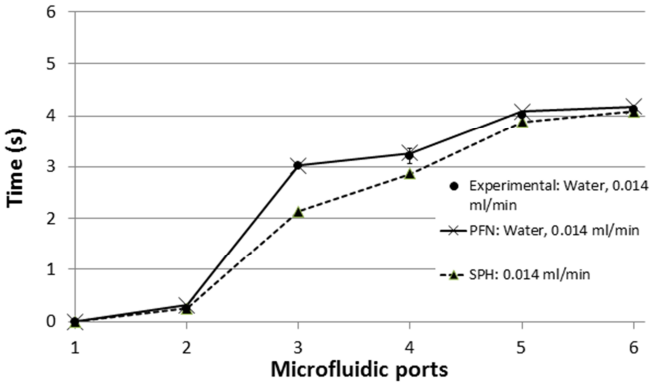

Fig. 18. Experimental, PFN and SPH results for constant flow.

The PFN simulation matches the experimental results with a lot of accuracy. In the SPH simulation, maintaining a constant pressure on the particles at the input of the fluidic network is hard to obtain in practice. Nevertheless, the PFN and SPH results coincide in an acceptable way.

B. Different pressures

Water was introduced using an external pressure source, implemented by a difference in height between the reservoir level and the outlet. Two different heights were used to check the simulation results, resulting on 200 and 400 Pa. Once the microfluidic device was filled by water, the reservoir filled by a mixture of water and Rhodamine B was connected, and the flow of the coloured liquid was observed by a microscope. As a result, the elapsed times obtained to reach the different ports were experimentally obtained. Results are shown in Fig. 19, and compared with simulation results.

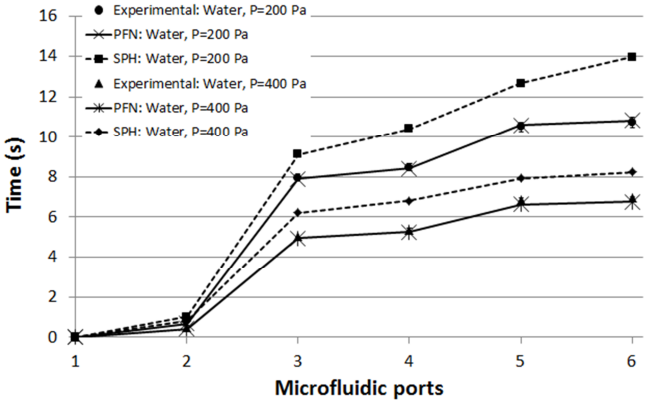

Fig. 19. Experimental, PFN and SPH results for two pressures.

Again, the PFN simulation is very accurate, matching with the experimental results. The SPH simulation is less accurate but it keeps in the order of magnitude of experiments and it behaves qualitatively as the reality because the liquid moves faster in the microchannels than in the microchambers, due to the first ones store less liquid than the second ones.

C. Different viscosities

The experiment was repeated using methanol, as a way to test a liquid with different viscosity (590 Pa s). In this case, Erythrosin B was used to colour the reservoir, as it dissolves much better in methanol. Results can be observed in Fig. 20, and compared with the results obtained by simulation.

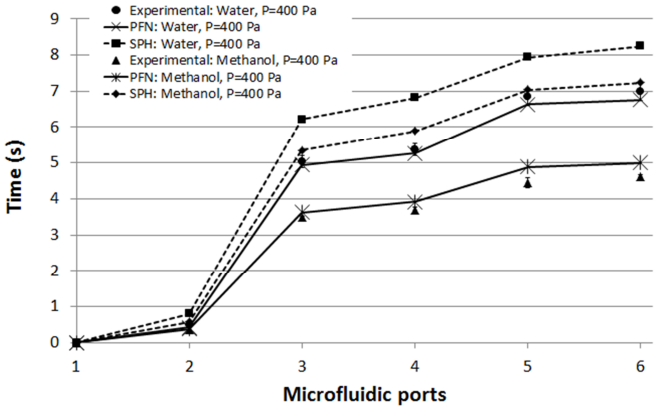

Fig. 20. Experimental, PFN and SPH results for two viscosities.

PFN results were very accurate. A maximum discrepancy of 9% was observed for methanol at the fifth microfluidic port. The SPH simulation keeps the order of magnitude and the qualitative behaviour.

D. Performance comparison with FEM

As an example of performance differences between the proposed simulation methodologies and another one based on FEM implementation, Table 1 compares modeling and simulation times for the proposed approaches and for ANSYS CFX [4]. Modeling time is an estimation of the time needed by an engineer to describe the microfluidic system with the different approaches. The simulation time is the time the tool needs to simulate the exposed experiment (after the experiment is described, that is, pure run time). Data are rounded as they express and order of magnitude.

TABLE 1. MODELING AND SIMULATION TIMES

TOOL	Modeling time (s)	Simulation time (s)
ANSYS CFX [4]	2700	1800
PFN	600	10
SPH	650	30

As can be observed, the proposed simulation mechanisms are much faster both in terms of modeling and simulation times.

V. CONCLUSIONS

The current paper presents a simulation methodology based on extensions to SystemC-AMS for the virtual prototyping of a pressure based microfluidic system. Such systems typically include a set of microfluidic channels and chambers for the processing of fluid experiments. The behavior of the liquids inside those cavities is modeled with the proposed extensions. SystemC being a standard language, widely used for embedded systems, the connection of the fluidic simulation with the embedded control software is very straightforward. This global simulation as well as the connection with other physical domains, also expressed with extensions of SystemC-AMS, is currently ongoing in the CATRENE CA701 H-INCEPTION project [10].

Two approaches have been developed for the fluidic emulation: PFN and SPH. The first one is based on the Hagen-Poiseuille analytical equations whereas the second one is based on numerical computations of a discretized model of the fluid.

The PFN modeling mechanism allows a simple geometry description of the fluidic system. However, the timing emulation results obtained were very precise over several types of experiments. Moreover, simulation approach demonstrated to be fast in terms of modeling and run time.

When it comes to the holistic modeling of a micro-fluidic system that takes into account the exact geometry of the fluidic network, it appears that the SPH simulation kernel is fast enough to offer the end-user quite approximate results in a reasonable time, compatible with the key idea that SystemC-AMS extensions are mainly dedicated to the building of first-order executable specifications that in turn aim at developing the embedded software as soon as possible with a quite faithful representation of all the hardware parts. As such, it is a good trade-off between the simpler (but quite accurate over a simple model of the geometry) PFN scheme and FEM.

Furthermore, considering fluids as a set of particles, the SPH approach have a lot of potential capabilities for the next steps in the emulation of a complex fluidic system. Processes like magnetic trapping of polarized cells, fluidic mixing or counting specific particles could be managed quite naturally.

ACKNOWLEDGMENT

This work has been supported by CATRENE CA701 H-INCEPTION project [10].

REFERENCES

[1] C. Ptolemaeus, Editor, “Sytem Design, Modeling, and Simulation Using Ptolemy II”, Ptolemy.org, 2014.

[2] www.modelica.org

[3] www.comsol.com

[4] www.ansys.com

[5] www.3ds.com

[6] www.mathworks.com

[7] www.plm.automation.siemens.com

[8] http://www.3ds.com/es/products-services/catia/capabilities/systems-engineering/modelica-systems-simulation/dymola/

[9] http://www.simulationx.com/

[10] https://www-soc.lip6.fr/trac/hinception

[11] ASI SystemC AMSWG, Standard SystemC AMS extensions language reference manual, version 2.0, Accellera Systems Initiative, 2013. [Online]. Available: http://www.accellera.org.

[12] IEEE Computer Society, 1666-2005 IEEE standard SystemC language reference manual, IEEE, Mar. 31, 2006, isbn: 0-7381-4871-7

[13] D. Mark, S. Haeberle, G. Roth, F.V. Stetten, R. Zengerle, “Microfluidic lab-on-chip platforms: Requirements, charateristics and applications”, Chemical Society Reviews, Vol 39, Issue 3, March 2010.

[14] Y. Zhao, K. Chakrabarty, “Design and Testing of Digital Microfluidic Biochips”, Springer, 2013.

[15] B. J. Kirby, “Micro- and Nanoscale Fluid Mechanics: Transport in Microfluidic Devices”, Cambridge University Press, 2010.

[16] H. Bruus, “Theoretical Microfluidics”, Oxford University Press, 2007.

[17] J. J. Monaghan, “Smoothed Particle Hydrodynamics”, Reports on Progress in Physics, Vol. 68, Num. 8, 2005.

[18] E. Hairer, C. Lubich, G. Wanner, “Geometric numerical integration illustrated by the Störmer/Verlet method”, Acta Numerica 12:399-450, 2003.

[19] Victor Fernandez, Elier Wilpert, Henrique Isidoro, Cédric Ben-Aoun and François Pêcheux, "SystemC-MDVP Modelling of Pressure Driven Microfluidic Systems", 2nd EUROMICRO/IEEE Workshop on Embedded and Cyber-Physical Systems-ECYPS’2014, Montenegro, June 2014.

***Biography of all Authors**
Click here to download Biography of all Authors: BiographyAuthors.pdf

Víctor Fernández received the M.S. degree in Physics (Electronics Section) at University of Cantabria in 1992 and the PhD in the same University in 1998. Assistant Professor from 1993 and Associate Professor from 2001 at University of Cantabria. During his period of Assistant professor he was also hired as digital microelectronic designer at Alcatel Espacio (Madrid). His research activities, carried out at the Microelectronics Engineering Group, covered Automatic Synthesis of Digital Systems, Channel Coding, On-Board Processing in Space Applications and Specification, Design and Verification of HW/SW Embedded Systems. In those areas, he is the author of several contributions to journals, books, conferences and patents.

Andrés Mena obtained the bachelor's degree in computer science engineering at Zaragoza University (Spain) in 2003. He then joined "Group of Structural Mechanics and Materials Modelling" at Zaragoza University where he obtained the MSc in Biomedical Engineering in 2012 and he is PhD candidate in Biomedical Engineering (Using Graphic Processor Units for the Study of Electric Propagation in Realistic Heart Models). He works in CIBER (Biomaterials and Tissue Engineering program) from 2010. In 2014 he joined AlphaSIP, where he focuses in the development of microfluidic virtual prototyping. He published 8 scientific papers and 3 contributions to international conferences.

Cédric Ben Aoun received his Bachelors and Master's degree in Computer Science from Sorbonne University Pierre et Marie Curie, Paris, France. Since 2014, he joined the LIP6 Laboratory in Sorbonne University as a Ph.D student.
His researches focus on the simulation of heterogeneous systems, virtual prototyping and composability of models of computation.

François Pêcheux is a full professor at Sorbonne Université Pierre and Marie Curie CNRS/LIP6 Laboratory in Paris, France. Its research activities focus on SystemC/SystemC-AMS based virtual prototyping, heterogeneous modeling, and on the development of a flexible simulation environment for multi-domain systems that relies on the seamless, synchronized cooperation of multiple models of computation. He coordinated for LIP6 several French and European projects like TSC, ADAM, WASABI, Beyond Dreams and H-Inception.
He is the author or co-author of multiple articles and conference contributions on (SystemC SystemC-AMS-based) IC design methodology for homogeneous and heterogeneous systems. He is responsible for the SystemC/SystemC-AMS training activities at Sorbonne Université.

Luis J. Fernández received the M.S. in Physics at Zaragoza University (Spain) in 2001. He then joined "Transducers Science and Technology" department at Twente University (The Netherlands), where he obtained the PhD degree in 2005 in the field of MEMS. During the next 5 years he focused on the development of Micro Total Analysis Systems (μTAS), microfluidic control elements and biosensors at Ikerlan (Spain). In 2010 he joined Zaragoza University (Spain), where he focuses in the development of microfluidic devices for biomimetic cell culture applications. He published 24 scientific papers, 52 contributions to international conferences, and holds 11 patents.

Separate Photo of Each Author

Click here to download Separate Photo of Each Author: BiographyAndPhotosAuthors.pdf

Víctor Fernández received the M.S. degree in Physics (Electronics Section) at University of Cantabria in 1992 and the PhD in the same University in 1998. Assistant Professor from 1993 and Associate Professor from 2001 at University of Cantabria. During his period of Assistant professor he was also hired as digital microelectronic designer at Alcatel Espacio (Madrid). His research activities, carried out at the Microelectronics Engineering Group, covered Automatic Synthesis of Digital Systems, Channel Coding, On-Board Processing in Space Applications and Specification, Design and Verification of HW/SW Embedded Systems. In those areas, he is the author of several contributions to journals, books, conferences and patents.

Andrés Mena obtained the bachelor's degree in computer science engineering at Zaragoza University (Spain) in 2003. He then joined “Group of Structural Mechanics and Materials Modelling” at Zaragoza University where he obtained the MSc in Biomedical Engineering in 2012 and he is PhD candidate in Biomedical Engineering (Using Graphic Processor Units for the Study of Electric Propagation in Realistic Heart Models). He works in CIBER (Biomaterials and Tissue Engineering program) from 2010. In 2014 he joined AlphaSIP, where he focuses in the development of microfluidic virtual prototyping. He published 8 scientific papers and 3 contributions to international conferences.

Cédric Ben Aoun received his Bachelors and Master's degree in Computer Science from Sorbonne University Pierre et Marie Curie, Paris, France. Since 2014, he joined the LIP6 Laboratory in Sorbonne University as a Ph.D student.
His researches focus on the simulation of heterogeneous systems, virtual prototyping and composability of models of computation.

François Pêcheux is a full professor at Sorbonne Université Pierre and Marie Curie CNRS/LIP6 Laboratory in Paris, France. Its research activities focus on SystemC/SystemC-AMS based virtual prototyping, heterogeneous modeling, and on the development of a flexible simulation environment for multi-domain systems that relies on the seamless, synchronized cooperation of multiple models of computation. He coordinated for LIP6 several French and European projects like TSC, ADAM, WASABI, Beyond Dreams and H-Inception.
He is the author or co-author of multiple articles and conference contributions on (SystemC SystemC-AMS-based) IC design methodology for homogeneous and heterogeneous systems. He is responsible for the SystemC/SystemC-AMS training activities at Sorbonne Université.

Luis J. Fernández received the M.S. in Physics at Zaragoza University (Spain) in 2001. He then joined “Transducers Science and Technology” department at Twente University (The Netherlands), where he obtained the PhD degree in 2005 in the field of MEMS. During the next 5 years he focused on the development of Micro Total Analysis Systems (μTAS), microfluidic control elements and biosensors at Ikerlan (Spain). In 2010 he joined Zaragoza University (Spain), where he focuses in the development of microfluidic devices for biomimetic cell

culture applications. He published 24 scientific papers, 52 contributions to international conferences, and holds 11 patents.