Emulation of Cyber-Physical Systems on FPGA

Maelic Louart[+*], Jean-Christophe Le Lann[+], Frédéric Le Roy[+],
Abdel Boudraa[*], Jean-Jacques Szkolnik[*]

[+]Lab-STICC, UMR CNRS 6285, ENSTA Bretagne, 29806 Brest cedex 9, France;
[*]IRENav, EA3634, BCRM Brest CC 600, Ecole Navale, 29240 Brest Cedex 9, France

Introduction

Cyber-physical system (CPS) context:

- CPS is a tight integration of computation with physical processes [1]. At a large scale a CPS is composed of several embedded systems interacting with each other in an environment.

CPS simulation problematics:

- CPS are heterogeneous systems mixing discrete and continuous aspects, software and hardware components and multiple disciplines (mechanical, electrical and software engineering);
- CPS modeling and simulation are a open problem: Complex modeling tool, long execution time.

Inspiration:

- Miller, Vahid, and Givargis who implemented on FPGA digital mockup of some human organs (lungs, heart, etc) to simulate their behavior (2011-2014) [2].

Our proposal:

- We simulate both CPS cyber and physical parts on a same FPGA.

Method

Our method comes from three **practical observations**:

- The quasi-systematic availability of highly domain-specific C/C++ codes able to simulate behavior parts of almost every CPS (physical, mechanical, chemical phenomena, etc.);
- The growing robustness of high level synthesis (HLS) tools able to synthesize hardware code from C++ code;
- The interesting characteristics of recent FPGAs (high computational power, large amount of resources, parallelism concepts easy to implement, and a native synchronization of computations).

Method application:

- Method divides CPS into components: one of them simulates CPS environment (physical part) and the other CPS' embedded systems;
- Dataflow directive implement FIFO or PIPO buffers between components with handshake protocol.

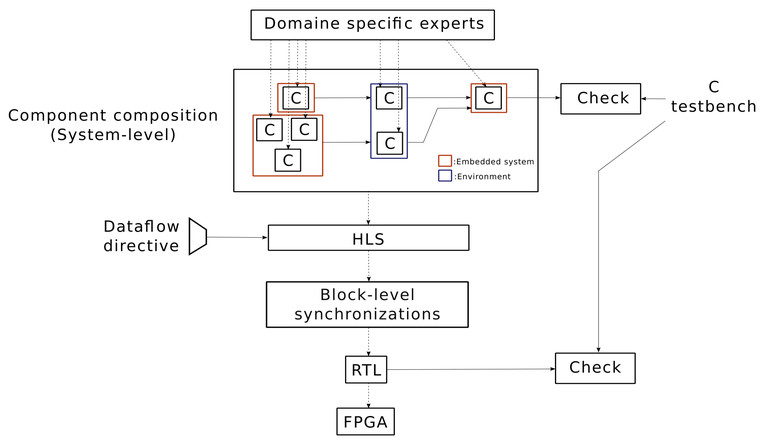


Figure 1: Overview of the approach for CPS simulator design

Limitation:

- Limited to directed acyclic graph.

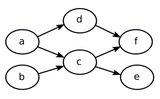


Figure 2: Directed acyclic graph

Application to automatic identification system

Case study:

- A set of vessels that move and exchange radio messages relying on a dedicated maritime protocol named automatic identification system (AIS).

Objective:

- Creation of a simulation framework to generate a wide variety of scenarios generating AIS signals within varied environmental conditions. AIS signals are tedious to be collected in real environment.

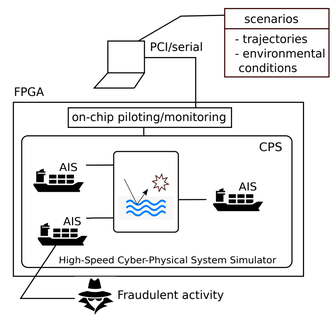
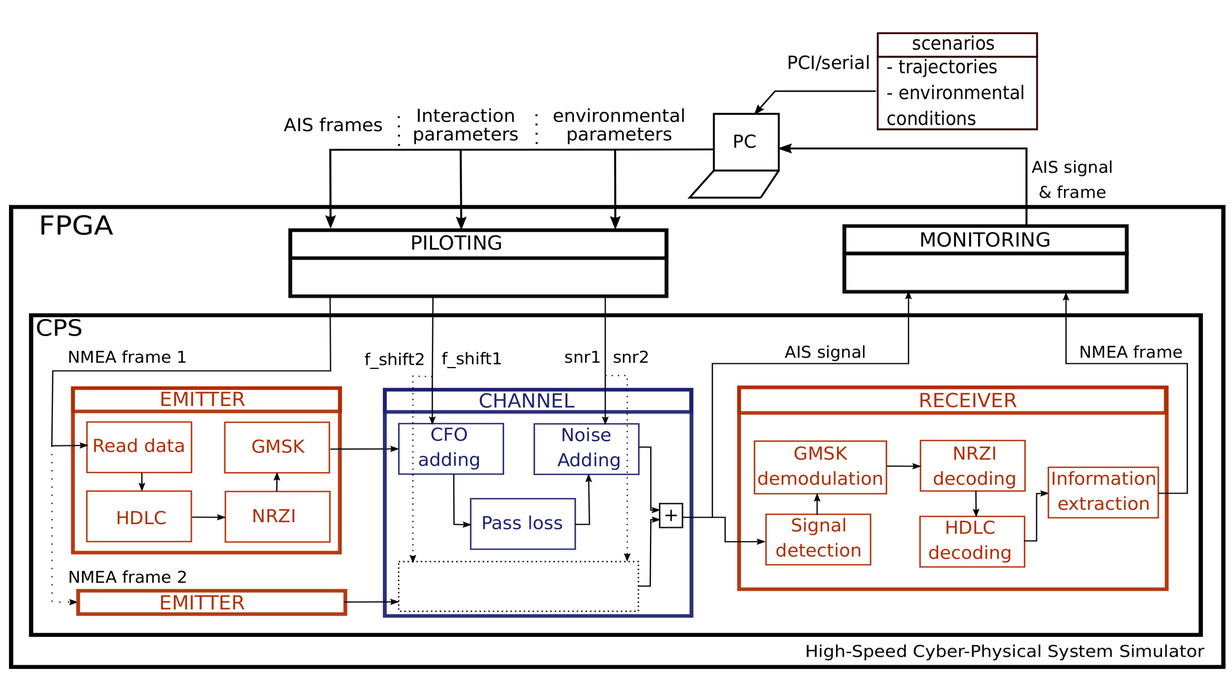


Figure 3: FPGAs as a AIS Simulation platforms

List of simulation framework components:

- **Emitter** that acts as an AIS transponder which, from an NMEA frame, sends a modulated AIS signal;
- **Channel** that simulates the environment to apply some physical laws to the AIS signals (SNR, carrier frequency offset, pass loss (Friis model: $\frac{P_r}{P_t} = G_r G_t \left(\frac{\lambda}{4\pi R}\right)^2$);
- **Receiver** that acts as a AIS transponder, which demodulates AIS signals to extract NMEA frame. It is an efficient prototype, it was tested on true baseband signals recorded near Brest;
- **Piloting** that allows a classical interaction with such a simulator (start, stop) and adjusts the simulated environment parameters;
- **Monitoring** that observes the dynamics of selected signals from the simulator.



: Detailed model of FPGA simulator

Synthesis and Simulation results

Synthesis results:

	BRAM	DSP	FF	LUT
Simulator	293(7%)	66(0%)	30508(1%)	41665(3%)
Emitter	97	1	8718	16357
Channel	97	13	1478	4312
Receiver	99	52	19904	20754

Table 1: Resources used by each component

Remark:

- Low FPGA occupency => Many Emitters can be simulated at the same time and environment can be made more complex.

Evaluation of the Receiver GMSK demodulation considering environmental conditions (SNR, CFO):

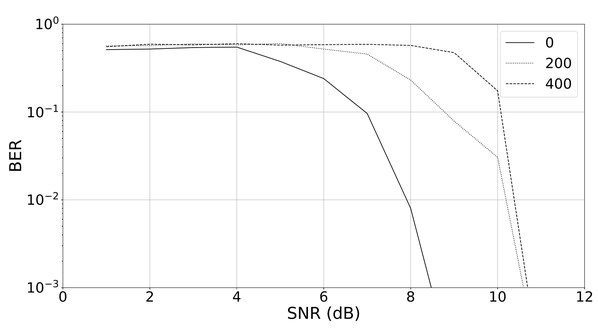


Figure 4: Bit error rate evolution related to SNR and carrier frequency error (CFO)

Simulation time results:

- During every simulation, 1000 messages are received by the *Receiver* successively from every *Emitter*.

Number Emitters	Software	Hardware	Gain	FPGA usage
1	576s	3.14s	183	7%
5	856s	3.14s	272	24%
10	1210s	3.14s	386	46%
22 (pred.)	2050s	3.14s	654	98%

Table 2: Simulation times obtained on FPGA and processor for various Emitters number simulated at the same time

- **Software**: a standard Intel Core I5 processor at 1.7 GHz with 16 Gb RAM;
- **Hardware**: a Xilinx Ultrascale+ HBM FPGA on a VCU128 board.

Remarks:

- Performance gain is already $\times 183$ compared to software simulation for only one *Emitter*;
- Performance gain was obtained without applying optimization directives to the code such as loop unrolling or pipeline which still leaves a speed-up margin for the FPGA simulation;
- Although the number of components *Emitter* was increased, the hardware simulation time remained the same => FPGA scalable characteristic.

Discussion

- Outside directed acyclic graph of architecture, Vitis HLS cannot synthesize CPS;
- Other high-level synthesis tools, allowing actor-based modeling without limit in the complexity of the architecture, using, among others pure dataflow semantics are to be experimented;
- Already Vitis HLS can be used to synthesize components, which are then exported as IP components to Vivado, to be assembled there according to the desired architecture.

Conclusion

- We demonstrate that FPGA can be seen as a plateforme of choice to simulate both cyber and physical parts of large CPSs;
- Speedup $\times 654$ compared to pure software simulation;
- The user can interact with the simulator changing the environmental parameters, monitoring the simulation (start, stop) and observing the signals exchanged by the components.

References

[1] EA. Lee and SA. Seshia. *Introduction to embedded systems: A cyber-physical systems approach*. Mit Press, 2017.

[2] B. Miller, F. Vahid, and T. Givargis. Application-specific codesign platform generation for digital mockups in cyber-physical systems. In *Electronic System Level Synthesis Conf. (ESLsyn)*, pages 1–6. IEEE, 2011.