



## BYTE PARADIGM: TESTING & DEBUGGING FPGA-BASED SYSTEMS

### 1 October 2008: Testing & Debugging FPGA-based Systems

INTERVIEWEE. Frederic Leens  
 Sales & Marketing Manager  
 TEL. +32 67 34 28 94  
 EMAIL. Frederic.Leens@byteparadigm.com  
 COMPANY. Byte Paradigm  
 WEB. <http://www.byteparadigm.com/>

**Q. First of all, tell us a little bit about yourself and your responsibilities at Byte Paradigm.**

A. I am founder and Sales & Marketing Manager of Byte Paradigm. Prior to creating the company, I worked during 6 years for many companies in Benelux and France, as ASIC, SoC, FPGA and digital board engineer for industries ranging from medical imaging to media and avionics. I hold a MSEE from the Mons Institute of Technology (FPMS, Belgium) and a Masters degree in Management. Today, I am in charge for sales and marketing at Byte Paradigm. While not purely technical any longer, my job involves a great deal of technology-related tasks – like studying customer requirements, detecting trends and defining new products.

**Q. Many of our readers may not be familiar with Byte Paradigm. Can you give us a very brief summary of the company, its products, and the “value proposition” that you bring to the FPGA community?**

A. Byte Paradigm was founded in 2005 by Olivier Rasmont, Didier Martiny and myself. We have about the same background, a common vision and one purpose: provide PC-based instruments to increase the productivity of embedded electronics engineers during design, test and debug.

To understand the idea beneath Byte Paradigm, let's compare FPGA and ASIC design. FPGA and ASIC or even SoC design have HDL coding (VHDL, Verilog, SystemC) and RTL / gate-level simulation in common. The complexity level in terms of gates is about the same. As you know, both for ASIC and FPGA, debug and test has become one of the most time- (and money-) consuming tasks in the design process.

What is unique with FPGA is that you can very quickly program a 'real' FPGA and that you can instantly change its functionality at no cost. This is a real advantage for debug and test – mainly because a real system always executes faster than a simulated system.

However, going from the 'design environment' on PC to the 'real environment' on a test board is not so straightforward. A test lab is often a separate environment, outside your familiar PC; real systems lack visibility and you often end up doubling your design time, because you need to design a second test system.-to test your system.

This is where Byte Paradigm's value proposition is: we want to help the engineer unleash the power and flexibility of FPGA during debug and test, by allowing him/her to go early on a real-world – or prototype – system. For this, he/she needs tools that offer a real flexibility for a wide variety of tasks – all about stimulation and observation – and a real continuity between the design environment and the test and debug environment.

Our current range of products features pattern generators, SPI and I<sup>2</sup>C exercisers and analyzers, and simple logic analyzers. They can all be controlled from PC and are programmable with common languages such as C; they are simply plugged into a USB interface and are very

compact, so any designer can put it next to his/her PC. In brief, they help the design 'talk' with their FPGA in the 'real world'.

**Q. Test and debug, while critical, doesn't exactly bring up happy thoughts among many FPGA designers. Tell us a little more specifically about your products and how they hopefully make the test and debug cycles easier, not for everyone but for FPGA designers in particular.**

A. Debug is hard because the FPGA-based systems are getting more and more complex. An FPGA-based system is often the work of several engineers. It is composed of custom firmware, third-party IP and many components taken off-the-shelf (memories, ADCs, DACs, processors). To tackle this complexity, a 'piece by piece approach' is often used: each functional piece of the design is validated separately. This gives the engineer a good insight about how each piece works in the real world. This may allow him/her to avoid pitfalls during integration.

For doing this, an FPGA designer basically needs something to send stimuli to an FPGA and something to observe the results of the stimulation. And he/she wants to do this easily, without having to design another board to do it.

That is what our current and future PC-based instruments help the FPGA designer do: generate stimuli from a PC and observe the FPGA from a PC.

If each piece of a design is validated and 'FPGA-proven', then it becomes easier to debug and test a fully assembled system. Piece-by-piece real world validation can also bring confidence in third party designs and IP, not to talk about validation.

**Q. Are your products limited to certain FPGA vendors (Xilinx, Altera, etc.) and/or certain product families within them?**

A. No, they are not limited: our current products interface FPGA through its digital I/Os, LVCMOS standard.

However, we are getting some new and more advanced products ready that may have some feature specifically developed for one given FPGA vendor in the future.

**Q. Most of the FPGA vendors provide their own development tools - tell us how your products complement these tools provided by Xilinx, Altera, Lattice, and others.**

A. Most of these tools make use of the JTAG port for observing what is going on inside FPGA. Our instruments access FPGA through its functional ports. In most cases, the two approaches are really complementary. Moreover, we think that debug and test is both about *observing* AND *stimulating*. This is also what we hear from our customers – this is why, we'll keep on focusing on both collecting data from FPGA and generating stimuli to FPGA in the future.

**Q. Many readers of this guide will not be designing FPGAs per se but rather working with a board vendor to create a semi-custom FPGA-based board. Are your products of value to these sorts of individuals as well?**

A. Sure. Our products act at I/O level. A simple example of non-FPGA use on board would be the stimulation of a DAC or interacting with a board through its I<sup>2</sup>C bus, for instance.

Actually, we initially thought of Byte Paradigm as a solution provider for FPGA debug and test. Our products revealed quite useful for board and ASIC / SoC designers as well and are notably

used today by fabless semiconductor companies for post-manufacturing prototype functional validation and qualification.

**Q. FPGA designs of all types often involve intellectual property (“IP”) as well as software from multiple vendors as well as from scratch by the designer. Do your products help in any way during the integration (and hence test/debug) of either the software or IP?**

A. Yes, and here is a concrete example for this. We recently offered support to a customer working in the digital video industry. They had to integrate a third-party IP together with a piece of firmware that they designed themselves. It did not work and they were stuck – not knowing which part of the design had a problem. Actually, some item of the IP specification was not well understood – matter of how data had to be exchanged between with the IP. The customer successfully communicated with the IP implemented in an FPGA with our pattern generator; the IP did what it had to do but, playing with the sequence of stimuli sent to the IP, they realized their own design did the things a little differently. It did not match the simulations. They tweaked their design a little and quickly succeeded to let the 2 pieces play together. They confessed the couple ‘FPGA’ plus ‘PC-instrument’ really provided the flexibility they needed to check this problem quickly and explore many stimulus variations.

This all is actually a piece-by-piece validation approach. It greatly helps speeding-up the full system integration. It’s just like you ‘secured’ partial designs before assembling them.

**Q. Many of the most troublesome test problems involve protocols. Tell us how your products can help with protocol test, even for some of the standard protocols that are “well established.”**

A. Actually, there may be 2 problems with protocols and FPGA, even well established:  
The first problem occurs when you design a given protocol master or slave port in your FPGA. This is a new design, and you want to test the port to check if it is compliant to the protocol – say, I<sup>2</sup>C or SPI. Using one of our protocol exercisers / analyzers or our pattern generators may help you test and debug your design. There are also specific problem using a protocol, when a popular protocol knows many variants. This is the case for SPI, for example, which has known many variations. Testing and debugging an ‘almost compliant’ port would need the appropriate environment. We strongly believe that PC-based instruments can offer this flexibility.

There is another case, where you also have a port using a given standard or custom protocol running in the FPGA and there is nothing wrong with it. However, this functional port can be a very handy ‘access point’ to stimulate and observe the rest of the design. This is what I call ‘debugging through a protocol’. Similarly, our protocol masters and pattern generators can help do this.

**Q. Would you like to tell us more about your future products?**

A. Well, we are getting several new products ready. Some have already been announced, like the followers of the GP-22050 in the GP Series USB devices. From mid-2009, we’ll be launching a new platform for pattern generation. This is an exciting full new development we have been busy with for months. I can’t tell you much about it, but for sure, it is going to offer more performance and features. Target market is high-end FPGA debug and test.

**Q. Thank you for this interview.**