All industrial/Student/hacker abstracts for Stockholm and Copenhagen, FPGAworld 2015

1 Title: The critically missing VHDL testbench feature - Finally a structured approach

Abstract: Verification is 51% of total FPGA development time. Even more for control or protocol oriented design.
Ignore this and you will iterate forever in the lab and suffer from bad product quality. The root cause is corner cases triggered by certain input and FSM combinations. Hitting a given corner case in normal simulation is very unlikely, - but in the final product it will hit you hard. A structured VHDL approach to verifying such corner cases has not been commonly available. UVVM (Universal VHDL Verif. Meth.) changes this picture completely.

From: Espen Tallaksen, <espen.tallaksen@bitvis.no>
Country: Norway and Bitvis

1.1 Tutorial, Unified VHDL Verification Methodology (UVVM)

From: Espen Tallaksen, <espen.tallaksen@bitvis.no>
Country: Norway and Bitvis

Abstract:
This tutorial will explain how UVVM may help you reduce your verification time significantly - and at the same time improve your product quality.

Corner cases in the specification and your implementation often result in lots of FPGA bugs. Most of these bugs are very difficult to detect in normal simulation or lab test, and very often quite a few of these bugs are not detected until experienced by the customer.
UVVM is a complete VHDL based verification environment platform that will cover all the most important aspects of FPGA verification. We will show how this works and how you can use this platform to make your own testbench and your own structured verification components based on the examples provided. As a simple example it took only one single hour to make a UART verification component from scratch (but based on a simple bus interface verification component and UART BFMs. This new UART verification component could then be used in a very structured and reusable verification environment.
Please note that this presentation assumes that you have attended the presentation 'The critically missing VHDL testbench feature - Finally a structured approach' in track A4 earlier today.
UVVM will be released very soon after FPGAworld.

2 Title: OSVVM for VHDL Verification

Abstract: Open Source VHDL Verification Methodology (OSVVM) is a comprehensive, advanced VHDL verification methodology that adds randomization (constrained and intelligent), and functional coverage. This presentation provides an in depth overview of these capabilities. Like UVM, OSVVM is a library of free, open-source code. However, OSVVM goes a step beyond other verification languages with its Intelligent Coverage methodology that is currently the only portable, language-based, intelligent testbench solution.

From: Jim Lewis, <jim@synthworks.com>
Country: USA and SynthWorks Design Inc

3 Title: OpenCL in an Embedded Environment

Abstract: With todays FPGA's that have ARM CPU's, comes the possibility to run OpenCL in an embedded environment. OpenCL gives an embedded SW developer the power of custom hardware acceleration without the need for writing low level HDL. It also enables a FPGA developer to be more productive in implementing algorithms. And because OpenCL code can be executed on a PC, implementations can be quickly evaluated and verified. This presentation will describe Alteras OpenCL solution for their SoC FPGA's.

From: Johan Karlsson, <johan.karlsson@xelmo.com>
Country: Sweden and Xelmo AB

4 Title: The FPGA on a Printed Circuit Board

Abstract: - Routing out from a «FPGA (BGA)» related to BGA pitch.
Design suggestions for BGA (0.8mm , 0.5mm and 0.4mm pitch)
- HDI Design rules
Different HDI via-structures, “ALIVH (Anylayer Inner Via Hole ”Stacked & Staggered mVias , Buried vias , Filled &
Capped vias Capabilities Decide IPC Class (2 or 3) before you start your layout
- Impedance requirements and considerations
- The importance of a net-list in PCB production

From: John Steinar Johnsen, <josse@elmatica.com>

Country: Norge and Elmatica AS

5 Title: Accelerating embedded software development and ASIC verification with FPGAs

Abstract: Two often underestimated use modes of high-density FPGAs are their viability as a verification tool for ASIC verification and as a development tool for embedded software development.
This paper provides some insight into why and how FPGAs have become an essential methodology in today's ASIC and SoC development.
The focus will be on ASIC-to-FPGA specific transformations like clocks and memories. The main use mode to be discussed is for providing a pre-silicon prototype for early, embedded software development.

From: Juergen Jaeger, <Juergen@cadence.com>
Country: USA and Cadence Design Systems, Inc.

6 Title: Automotive ADAS feature saves lifes with flexible HW (FPGA etc.) – practical implementation reference.

Abstract: Automobile accidents are currently killing 1.25 million people per year worldwide and that figure is rising.
The new mantra is "avoiding collisions."
Consumer surveys show that people want safety features in cars, that these safety features sell cars, and that interest in ADAS features is actually higher than for any other electronics-based automotive category.
FPGA enables ADAS systems, saving lifes in the traffic. A reflection of present system.
From: Tryggve Mathiesen, <tryggve.mathiesen@qamcom.se>
Country: Sweden and Qamcom R&T AB

7 Title: High-speed FPGA-based stereovision system - a success story

Abstract: The presentation concerns an industrial stereovision system based on the Xilinx Zynq 7020 FPGA SoC which combines Linux on ARM and with surrounding reconfigurable logic that works as a hardware accelerator. The system is capable of working in real-time with ~50 FPS in VGA and above 65 fps in lower resolutions. The architecture includes sensor control logic, demosaicer, rectifier and stereovision cores.

From: Rafal Kapela, Ph.D., <rkapela@antmicro.com>
Country: Poland and Antmicro Ltd/Poznan University of Technology

8 Title: New Low Level C programming book for students with Altera BeMicro 10

Abstract: In an embedded computer system, resources are in general very limited and this is particularly true for memory (data and program memory). C code for embedded systems are characterized by small footprints, bit manipulations of variables, intimate hardware interaction and typical projects are focused on API design, i.e. writing device driver code for user applications. The textbook "Low Level C programming" treats the subject of embedded system programming with a mixture of theory and training (hands on). The training are focused on a non-expensive development board from Arrow

(BeMicro MAX 10) and the necessary configuration files (about 10 different) for the FPGA can be downloaded from webpage for the different training.

From: Lars Bengtsson, <lars.bengtsson@physics.gu.se>
Country: Sweden and University of Gothenburg

9 Title: Real-time operating system, hardware or software?

Abstract: One important purpose of a real-time operating system is to present a result in right time. The question is how much time can be saved using a real-time hardware OS accelerator in hardware, compared to today's software kernel in a FPGA device with NIOS processor? The hardware based real-time kernel is designed in VHDL with a thin software device drivers. The software kernel used is FreeRTOS and the hardware kernel used is Sierra with exact same hardware architecture.

From: André Norberg, <andre.norberg@agstu.com> (System Designer)
Country: Sweden and AGSTU

10 Title: The IP-component I2C_Master_IP

Type: student/hacker

Abstract: An I2C Master IP-component based on VHDL-code from eewiki.net has been developed with Altera's tools. The majority of the logic is implemented in hardware, the software driver basically consists of simple read and writes. Additionally a software library with functions for communicating with the Digital Temperature Sensor MCP9808 has been developed. An example application is demonstrated with Altera's development board DE2-115.

From: Lars Högberg, <larhog@gmail.com>
Country: Sweden and AGSTU School