Fast iteration and prototyping in HPC medical applications: a case study with Mentor Vista

Background and objectives

Architecture design and prototyping on low-power image processing platforms.
- Medical X-Ray imaging application.
  - Improve performance per watt.
  - 10x by 2018, 200x by 2023
- Reduce the level of radiation by 75%.
- More powerful image processing.
- The device needs to be small.
- Processing close to the sensor.

Use of tools for fast exploration on new heterogeneous architectures for high-performance computing (HPC) applications.
- Fast performance evaluation of new chips and platforms.
- Architecture-adequacy discovery for domain representative algorithms.
  - Image processing.
  - Pixel tracking.
  - Big data.

Case study

Image processing kernel chain.
- Reference implementation in sequential C code.
- Requirements:
  - HW/SW co-designers.

Tools available.
- HDL/ VHDL
- SystemC
- Model-checking
- HW/SW co-debugging tools

Must: 1024x1024 25 fps 420 Mbits/sec
Want: 1344x1344, 30 fps 870 Mbits/sec

Different data access patterns.
- Pixel by pixel.
- Sliding window.
- Chessboard.

Modeling the HW with Vista™

SystemC and TLM environment.
- Virtual prototypes of SoCs.
- Altera A10, S10, Zynq, Freescale IMX6, etc.
- Model-builder for defining the HW.
- ISS for the CPU cores.
- Specify the IPs in the FPGA fabric.
- Choose the communication protocols (e.g. AXI).

IP modeling and specification.
- HW registers.
- Memory addresses.
- IRQs.
- Energy consumptions.
- Physical bus connections.

Bus mapping manually defined.
- Design the memory layout of the FPGA.

Kernel drivers implementation for the FPGA IPs.
- UIO-based approach.
- Modify the Linux device tree and rebuild root.
- Forward the memory, bus, and IRQ mapping defined in the previous stage.
- Possible UIO driver modification required.

Developing the SW with Sourcery™

C/C++ integrated IDE for embedded devices
- SystemC and TLM environment.
- Altera A10, S10, Zynq, Freescale IMX6, etc.
- Model-builder for defining the HW.
- ISS for the CPU cores.
- Specify the IPs in the FPGA fabric.
- Choose the protocols (e.g. AXI).

HW/SW co-debugging.
- Simultaneous debugging of HW and SW.
- Vista for the HW.
- Sourcery for the SW.
- Direct connection to the virtual prototype.

Architecture space exploration

Migration from Arria10 to Zynq.
- HW FPGA models almost the same.
- SW application code totally portable.
- Change the bus sizes of the FPGA HW models.
- Information provided by Mentor Vista’s compiler.
- Change bus sizes of the fabric or use adapters.

Find the corresponding bus between FPGA fabric and SoC.
- Learn about the underlying architecture.
- Choose a new bus to connect.
- Recalculate memory mapping.

Redo the DTS file for the new platform.
- Requires deep understanding of the platforms and their DTs.
- Not very different devices (A10 vs Zynq).
  - Both of them SoC.
  - Waiting for the Zynq Ultrascale.
  - Still untested real heterogeneity.
  - GPGPUs unexplored.

Conclusions

Very mature tools.
- Ready to be used in real developments.
- Still some level of expertise is required.

Intended to help the HW/SW co-integration using a virtual prototype.
- Not focused on fast iteration and early (real) prototyping.
- Best use case: software application on SoCs.

Share the expertise about the tool in order to speed-up the learning curve.
- Generic HW models.
- Bus mappings.
- Application templates.

The research leading to these results has received funding from the People Programme (Marie Curie Actions) of the European Union’s Seventh Framework Programme (FP7/2007-2013) under REA grant agreement No. [517446] H2020 “Intelligent Fast Interconnected and Efficient Devices for Frontier Exploitation in Research and Industry”.

Alberto García-Fernández
(alberto.garciafernandez@thalesgroup.com)