FLOSS Tools for High Level Synthesis

Integrating the FPGA into the Operating System

Javier D. Garcia Lasheras
GL Research (Spain)
jgarcia@gl-research.com

FOSDEM 2017

State of the Art

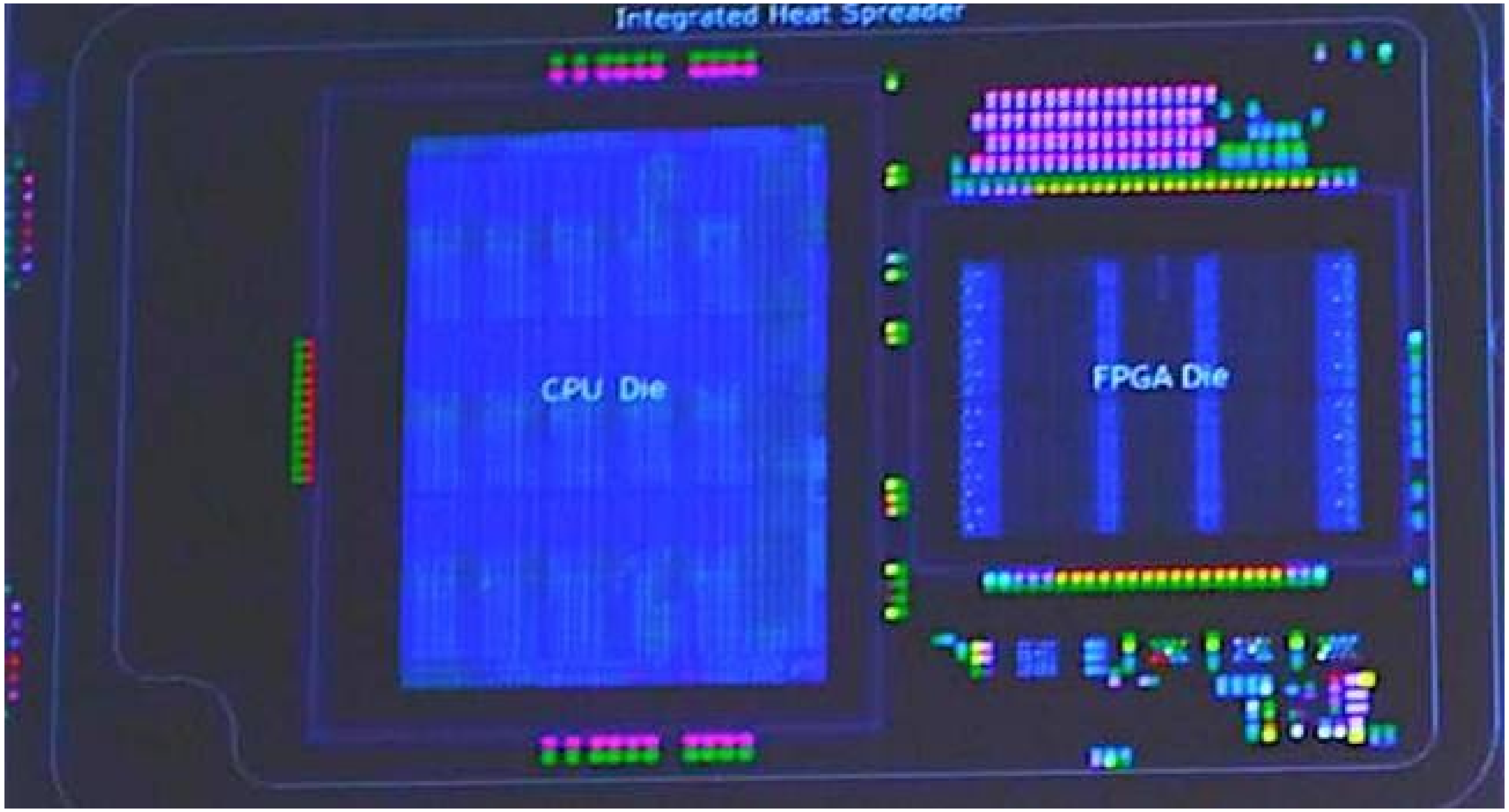
- A new generation of algorithms demanding for extreme computational power have recently arrived:
  - Embedded speech & vision (face & voice recognition)
  - Deep & Machine learning
  - Big-Data and data-mining
  - ...
- Von Neumann CPUs improvement driven by Moore's Law is not scaling anymore.
- **New architectures & software required!!!**

FPGA to the rescue

- GPU tools (e.g. Nvidia CUDA) are very powerful, but some problems don't benefit from massive Floating Point parallelism, .e.g. inference in artificial vision and pattern recognition performs better in fixed point under the main metrics:
  - Silicon Real-State
  - Power Consumption
  - Algorithm Accuracy
- Flexibility is a must:
  - Software evolves fast, but custom ASICs cannot change.
- **The whole industry is looking at FPGAs for answers!!!**

“Wintel” strikes back

- Intel acquires Altera (e.g. Xeon + FPGA below)
- Microsoft's announces **Project Catapult**
  - Bing Search Engine, Office365...



Picture source: https://www.theregister.co.uk/2016/03/14/intel_xeon_fpga/

And they are not the only ones...

- Baidu accelerates SQL at datacenter scale:
  - Xilinx Kintex Ultrascale KU115
- Amazon Web Services EC2 (F1):
  - Up to 8 Xilinx Ultrascale+ VU9P per instance
- IBM and “Intelligence Augmented”:
  - PowerPC + Xilinx FPGA
  - TrueNorth (designed by Cornell's FPGA experts)

How High Level Synthesis works

- **TARGET**: accelerate software algorithms by using FPGA

- Automatically generate the code from a high level software:
  - OpenCL is becoming the de-facto standard.
  - GPU-like programming experience.

- Complex and expensive commercial tools:
  1) Profile the application by running on a Valgrind-like box.
  2) Automatically generate HDL code for the most intensive tasks.
  3) Compile the software and synthesize the generated gateware.
  4) Provide a driver infrastructure for FPGA loading/unloading

Our approach to FLOSS HLS

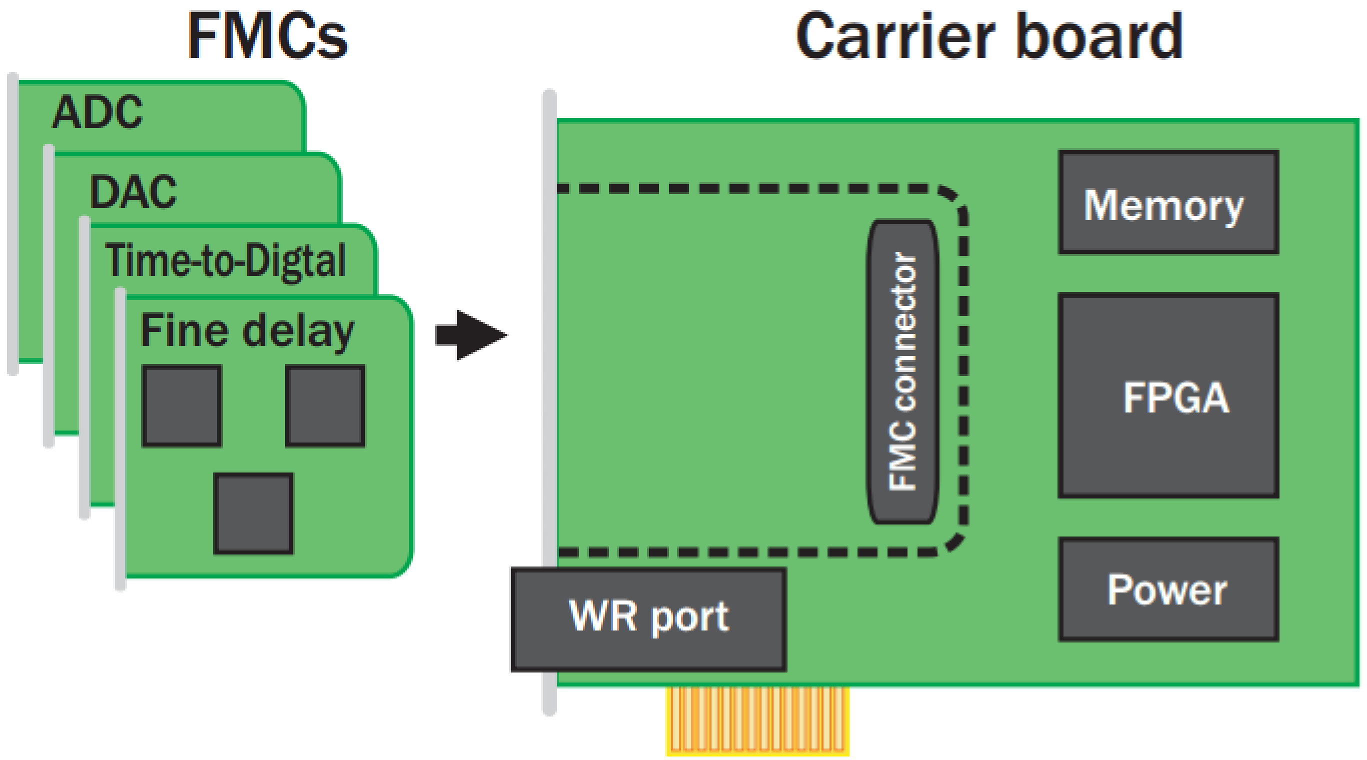
- Provide a set of HW accelerated libraries for specific purposes:
  - Digital Signal Processing
  - Embedded Vision
  - Inverse Kinematics
  - ...
- The provided libraries must include:
  - A software side providing a wrapper (classic API) to link with software.
  - A gateware side providing the optimized HDL code.
- **Transparency**: the software programmer must be able of using these HW accelerated libraries in the code just as standard ones:
  - using “includes” and calling “functions” to employ the code
  - system is on charge of coordinating data transient between SW and GW

Integrating the FPGA into the O.S.

- **Note**: we are only focused on Linux!
- Each of the HW accelerated libraries have their own associated bitstream.
- Different library versions may require different bitstream versions too.
- Package the associated bitstream in a Linux friendly way:
  - .deb, .rpm, .ipk...
  - Introduce dependencies with the software parts.
  - Pre-synthesized bitstream!! – only x86 processors are able to run synthesis!
- **Yocto Project** as building system
  - It allows for custom “distro” creation
  - We use **HDLMake** to provide YP recipes
  - Some examples next...

Example 1: CERN OHR FMC Kit

- http://www.ohwr.org/projects/meta-spec
- CERN's Simple PCIe FMC Carrier
- Versioned Kernel Module + UserSpace + FPGA



Example 2: Robotic Controls

- http://rosindustrial.org/
- ROS Industrial is working on providing intelligence to industrial robots:
  - ROS stands for Robotic Operating System
  - A collection of FLOSS libraries and tools
- There is a need for massive computation power!!!
  - GL Research is on charge of studying FPGA application on ROS-I.
- Candidate libraries:
  - Inverse Kinematics (IK)
  - Point Cloud (PCL)

The missing feature

Pareto's Rule (80-20%) applies to HW acceleration:
  - Not all the libraries are used at the same time/place.
- We would need a cache-like mechanism applied to FPGA configuration memory:
  - Bitstream “chunks” loaded/unloaded on demand.
- In general, we can just reprogram the whole FPGA
- Some FPGAs support partial reconfiguration:
  - Higher End Devices
  - Extra (and very expensive) licenses required!!!

Some conclusions

- **There are not FLOSS HLS tools around!!!**
- HDLMake allows for convenient management at an Operating System level.
- Need for a HDL library standard:
  - IP-XACT?
  - Alternatives?
  - FuseSoC as IP integrator?
- Workaround for partial reconfiguration?

Bonus Track: Introducing HDLMake

- An assistant tool for HDL developers:
  - Command line tool
  - Web based GUI (under development).
- A very “Pythonic” tool:
  - Written in Python
  - Allows for Python code injection at Syn/Sim time
- It automatically generates:
  - Synthesis Makefiles
  - Simulation Makefiles
- Multiple FPGA vendors and tools supported.
  - Commercial & FLOSS
- http://www.ohwr.org/projects/hdl-make

PLAY