Kactus2: Open Source IP-XACT tool
Current research

SoC design methods and tools
Kactus2 open source
IP-XACT tool

Parallel video encoder implementations
DSP, FPGA, Multicore
Kvazaar open source
HEVC encoder
Neural Network & Genetic Algorithm computer TUTNC (1996)

7xFPGA  4xDSP
"HIBI" on-chip network
3x 10xNIOs (2005)
Kvazaar open source HEVC encoder on CycloneV SoC-FPGA (DSD’2015)

INPUT

- Kvazaar 0.24
- Untimed SystemC
- Timed SystemC
- Generic HAL code
- Benchmark code
- Drivers source
- Kvazaar C models from timed SystemC model
- HW library
- Linux Yocto build
- Executables

DESIGN STEP&TOOL

- Gprof
- CatapultC
- SystemC
- Measurements
- ARMGCC
- CatapultC
- Precision synthesis
- Qsys
- Quartus
- Image tools
- Terminal

OUTPUT

- Performance estimation
- Area estimation
- HW/SW partitioning Exploration
- FPGA Performance annotation
- Executables, drivers
- Verilog
- Netlist
- Raw binary (FPGA config)
- Preloader
- Device tree blob
- U-boot bootloader
- Linux Image
- SD card image
- Load and run HEVC@SoC-FPGA

Video decoding@PC

Compressed video stream over Ethernet to PC

Camera

HEVC live video encoding on CycloneV SX@VEEK-MT-C5SoC

9.10.2015 TDH
Design methods & tools roadmap

SDL
- SDL-based DSP+FPGA design for custom WLAN MAC

UML
- UML2.0 based MP-SoC video+WLAN

IP-XACT
- Kactus1 tool
- IP-XACT 1.2
- Kactus2 IP-XACT + extensions
- ARM + FPGA multicore

HLS
- Kactus3D
- Kactus2 3D
- Kactus2 IP-XACT

IP-XACT Motivation
ASIC/SoC-FPGA design challenges

- Multitude of tools, languages and specification styles
  - Different formats (syntax)
  - Meaning (semantics)
  - Intention (how language or tool is applied—or abused...)

- Abstractions above RTL must be used for design space exploration

- Design for deadline -> design for reuse

- SW development should start in parallel with HW design
The HDL challenges

- VHDL/Verilog HDLs include three aspects mixed
  - Structural and behavioral descriptions
  - Control for configuration (functions, param. propagation, generators, conditions)
  - Virtual libraries (name based references)

- Implicit references that get evaluated late/somewhere in the design flow

- Vulnerable to errors if 100k files and multiple vendors
  - Wrong path/files, conflicts in (re)naming, scripts dependent on file version, ...

- Coding style agreements does not seem to help

5626 pages for specification!
The IP-XACT Story

- The idea was published in DAC’2003 conference
- Mentor’s closed tool format + Philips’ IP assembly tool = IP-XACT 1.0
- Now IP-XACT is the IEEE standard 1685-2014

Another aspect of SPIRIT is the rationale for it coming into existence. Although the first six members included the three major EDA tool vendors, two semiconductor companies in Europe and one IP company, it was really motivated originally by Philips Semiconductor’s wish to create a tool for SoC design via IP integration called NXPBuilder and Mentor Graphics’ desire to sell their Platform Express technology to Philips as a basis for NXPBuilder. Platform Express was provided for free for individual use and they tried to sell the technology to companies who wanted to build IP integration flows. It relied on an XML format for IP metadata.

Philips Semiconductor did not want to build their IP integration flow on top of a proprietary Mentor Graphics IP metadata format and thus be trapped into single supplier support; it insisted that the Mentor format become the basis for an interoperable IP metadata format. Mentor signed up ARM as an important IP supplier to Philips Semiconductor, and to gain the credibility of the largest source of independent IP in the industry. Neither Cadence nor Synopsys had ESL IP integration tools at that time but thought that it might be an important new market. Given the common European interest in standards-based design processes and tools, and given...
IP-XACT

- Standard
  - IP-block model
  - SoC design model
  - Integration and configuration flow
  - Tool interfaces

- IEEE1685-2014 include
  - 24 XSD files
  - 790 elements, 241 attributes
  - Vendor extensions
    - E.g. Xilinx adds ~200 elements

- Attempts to be a methodology, not as yet another exchange file format
Kactus2 motivation

- Launched from the needs of embedded system SMEs – and need to re-use the results of student projects
- Subcontractor SMEs need (affordable) tool to packetize their IP for integrator companies
- Thus:
  - Users may not be IP-XACT experts
  - Tool must have much better usability than EDA tools on average
  - Tool must be easy to install and ready to use right from the start
  - We keep every release as stable as possible even if it lacks features
Scope of IP-XACT

- Model for IP/design exchange and reuse
- Placeholder for *generators* configuring/affecting designs
- Not a language nor e.g. definition of automation scripts
IP-block = The IP-XACT component

- IP-XACT component is a structural model of the IP-block
- Several implementations at different abstraction levels and languages can be included as Views and FileSets
  - All implement the same external interface
- Reusable: self-containing, explicit definitions, parameters must be evaluated within the component

Common structure and definitions:
- Interfaces
- Parameters
- Registers (for SW)
- (Non-functional properties)

Analogy: VHDL Entity = interface

VHDL Architecture = functionality

Associated files for
- Behavior
- Documentation
- Whatever out there but the interface

Traditional file reference

IP-XACT component

FileSets
The IP-XACT Design

- **IP-XACT design** is a model of the SoC
  - Component instances
  - Interconnections, formalized by **Bus definitions**
  - Design-wide and instance specific component configurations by **Configurable Element Values** that are specific to **Design Configurations**

- Parameters propagate only one level downwards in hierarchy
  - Component must always be independently reusable independent of context
IP-XACT hierarchy

- **C**: Component
- **DC**: Design configuration
- **D**: Design
- **CI**: Component instance

 hierarchical reference (VLNV)

These are e.g. different product versions with different active Views on instantiated components
IP-XACT Design Flow

- Structure: IP-XACT description
- Behavior: source HDL files
- Automation: IP-XACT generators
- All design configurations take place at IP-XACT level, not at HDL level
  - Generated HDL code is “simple”
Kactus2 Scope

- Import wizards for VHDL, Verilog and Quartus pin map
- Editors for
  - Component creation
  - Design capture & configuration
- Generators for VHDL, Verilog, C (headers for registers), SystemC, PADS PCB design, HTML
Challenges in applying IP-XACT

- Standard does not ensure compatibility of bus/abstraction definitions from different vendors
- Vendor extensions complicate exchangeability
- Re-generation of HDL code is problematic if it was manually fixed in between (general problem)

Common problems:
- User edits IP-XACT XML files manually and add comments to anywhere in the XML
- Try to implement over-generic legacy HDL projects in IP-XACT as such
- Make ruthless file/name dependencies and does not respect the IP-XACT way
- Abuse IP-XACT elements, e.g. a parameter is NOT for a file path reference or for controlling product versions
Kactus2 tool
Kactus2 demo

- Component ports, address space, memory map
- Import from VHDL file
- (draft a new component and generate stub VHDL code)
Kactus2 tool project
Kactus2 Project

In a Nutshell, Kactus2...

... has had 1,843 commits made by 10 contributors representing 233,277 lines of code

... is mostly written in C++ with a very well-commented source code

... has a codebase with a long source history maintained by a average size development team with increasing Y-O-Y commits

... took an estimated 60 years of effort (COCOMO model) starting with its first commit in October, 2011

- Language: C++/Qt
- SDK: VisualStudio
- Runs on Win, Linux, (Anrdoid)
- Key contributors at TUT
- Requirements, feedback, support provided by companies
Kactus2 Releases

- Three main waves of development
- 3.0 upgrades to 1685-2014 (Oct 2015)
- 8k+ downloads
Call for contributions

Kactus2 core development

Kactus2 plugins

http://kactus2.cs.tut.fi
http://funbase.cs.tut.fi
kactus2@cs.tut.fi

Best practise design examples

Coding camp for your design?
We are pleased to help you - maybe get together and code together?

http://kactus2.cs.tut.fi
http://funbase.cs.tut.fi
kactus2@cs.tut.fi