FPGA Tooling Interoperability with the FPGA Interchange Format

CHIPS Alliance Fall Workshop, 2021-10-12
Maciej Kurc, mkurc@antmicro.com
OPEN SOURCE FPGA TOOLS

- We have been observing continuous development of open source FPGA tools over the past few years (and actively contributing)
- Together many of the tools allow to assemble a complete HDL-to-bitstream implementation flow
- Some of them can be mixed with existing proprietary FPGA vendor tools

- Yosys
- nextpnr
- VPR
- ODIN II
TYPICAL FPGA TOOLCHAIN FLOWS

Single toolchain:

```
HDL design → Synthesis → Placement → Routing → Bitstream generation → Bitstream
```

Separate synthesis and PnR:

```
HDL design → Synthesis → Placement → Routing → Bitstream generation → Bitstream
```
The problem: limited interoperability
HETEROGENEOUS FLOWS HAVE NOT BEEN POSSIBLE
PROBLEMS WITH INTEROPERABILITY

- Different data formats and/or their "flavors"
- Different data representation
THE FPGA INTERCHANGE FORMAT
THE MOTIVATION AND GOALS

- Antmicro with Google, Xilinx and other partners aims at enabling interoperability of existing FPGA tools and any new to come
- This can be achieved by providing a strict and consistent data representation and storage format
- Every tool supporting the FPGA Interchange Format should be just that: interchangeable with any other tool that provides the same functionality
THE COMPONENTS OF THE FORMAT

• Three major components:
  ▫ Logical netlist
  ▫ Physical netlist
  ▫ Device resources
LOGICAL NETLIST

• Carries information about logical structure of a design
• Contains:
  ▫ Logical cell libraries
  ▫ Cell instances and connections between them
  ▫ Cell attributes and parameters
• Similar in structure to the EDIF format but strictly defined via the schema
PHYSICAL NETLIST

- Stores cell placements and net routes within the FPGA fabric
- Contains:
  - Cell placement
  - Net routes
DEVICE RESOURCES

- Provides the complete description of the FPGA fabric architecture
- Contains:
  - Logic resources:
    - Tiles, sites and BELs types and instances
  - Routing resources:
    - Wires, nodes, PIPs
  - Supplementary information:
    - LUT element definitions
    - Timing data
    - and much more...
INTERCHANGE FORMAT SPECIFICATION

• The format is described via a set of schemas defined using the Cap'n'Proto library schema language.

• The library was chosen as it allows for generating consistent data access API while supporting multiple programming languages and being cross-platform.

• All the schemas are available in the GitHub repository:
  https://github.com/chipsalliance/fpga-interchange-schema
UTILITIES

Several utilities are also available that operate on the Interchange Format, including:

- Reading and writing routines to/from Interchange Format from/to Pythonic object models
- Sanity checking utilities for the Interchange Format data
- Conversion to/from Cap'n'Proto to a textual format (JSON, YAML)
- Conversion from the JSON output format of Yosys to the Interchange logical netlist

Utilities are written in Python and available on GitHub: https://github.com/SymbiFlow/python-fpga-interchange
TOOLS THAT ALREADY USE FPGA INTERCHANGE

Xilinx RapidWright

• The first vendor utility which takes advantage of the FPGA Interchange Format.
• Allows to write device resources for most of the 7-series and UltraScale fabrics
• Enables conversion of logical netlist and physical netlist to/from the DCP format readable by Vivado
• GitHub: https://github.com/Xilinx/RapidWright

nextpnr

• Enables full-fledged placement and routing using the FPGA Interchange Format
• Accepts device resources as the fabric definition
• Reads logical netlist as synthesized design input
• Writes physical netlist with placement and routing information as output
• GitHub: https://github.com/YosysHQ/nextpnr
ONGOING EFFORT AND FUTURE PLANS

VPR

- Development effort is in progress to enable the tool to use the FPGA Interchange Format
- Expecting support for logical netlist and device resource reading to be finished first

Yosys

- Expecting to have at least plugins which would serve as frontend and backend for the logical netlist
THANK YOU FOR YOUR ATTENTION!