The EPFL Logic Synthesis Libraries in Action:
A Development Snapshot of *mockturtle* & *tweedledum*

Heinz Riener and Bruno Schmitt
EPFL, Lausanne, CH
EPFL Logic Synthesis Libraries

Collection of open-source SW libraries for logic synthesis applications:

- Two years development
- Funded by EPFL’s Open Science Fund
- 9 open source libraries in C++
- Modularity & composability

Two “flagship” libraries:

- *mockturtle*: a C++ logic network library
  ([https://github.com/lsils/mockturtle](https://github.com/lsils/mockturtle))
- *tweedledum*: a C++ quantum compilation library
  ([https://github.com/boschmitt/tweedledum](https://github.com/boschmitt/tweedledum))
EPFL Libraries as Research Enabler

Modular software libraries in C++:

- State-of-the-art implementations
- Well-documented & -tested
- Header-only: easy to integrate

Users can tackle complex research problems

Logic optimizations are used to outside of hardware design:

- Pre- and post-processing in constraint solving
- Simplifying circuit verification problems
- Analyzing hardness of cryptographic circuits/logic encryption
- Optimization in high-level synthesis (or reactive synthesis)

mockturtle: a C++ logic network library

Flexible logic network library for research

- Inspired by C++ concepts
- Composability: Uses template metaprogramming to decouple implementation of logic networks from algorithms
- Views: Add or remove methods to a logic network
Scalable Generic Logic Synthesis

- "Scalable" = Similar to Mishchenko & Brayton 2006, but not limited to AIG
- "Genericness" = Network type in all algorithms is a template parameter
  - Syntax: methods exist and get the right parameters (checked by compiler)
  - Semantics: methods implement the expected semantics (network interface API)

Agnostic of the gate type
Can be implemented once and will work for all network types that provide the methods:
- foreach_input
- foreach_gate
- foreach_fanin
- foreach_output

Open-Sourced in GitHub

mockturtle has been public since April 25, 2018

- Link: [https://github.com/lsils/mockturtle](https://github.com/lsils/mockturtle)
Improved Robustness and Testing

- Coverage analysis and testing
- More benchmarks, fuzz testing, delta debugger?
- Bugfixes (based on GitHub contributors): Marcel Walter, Bruno Schmitt, Walter Lau Neto, Jinzheng Tu, Sahand Kashani, Max Austin, Jovan Blanusa, Giulia Meuli, ...
Research Contributions

- Optimization algorithms of majority logic networks (Eleonora Testa)
- Exact synthesis of circuit structures (Winston Haaswijk)
- Logic optimizations for XMGs (Zhufei Chu and Shubham Rai)
- Multiplicative complexity and multiplicative depth analysis (Mathias Soeken)
- Technology and logic mapping (Alessandro Tempia Calvino)
- Logic optimization for superconducting technologies (Dewmini Marakkalage)
- Generalized Boolean resynthesis (Siang-Yun Lee)
- Buffered networks (Siang-Yun Lee)
- ...

...
Future Research & Challenges

Improved scalability:
- Better quality-of-results with equal resource budget

Concurrency & parallelization:
- New algorithms to take advantage of modern hardware
- Support of modern C++ for concurrency enables new compiler optimizations

Flow scripts for *mockturtle*

Better debugging and testing infrastructure

Better support for nano-emerging technologies
The EPFL Logic Synthesis Libraries in Action:
A Development Snapshot of mockturtle & tweedledum

Heinz Riener and Bruno Schmitt
EPFL, Lausanne, CH