MAX PLANCK INSTITUTE
FOR SECURITY AND PRIVACY

The Threat of FPGA Reverse Engineering

Simon Klix & Nils Albartus

MPI-SP, Bochum, Germany

Recap

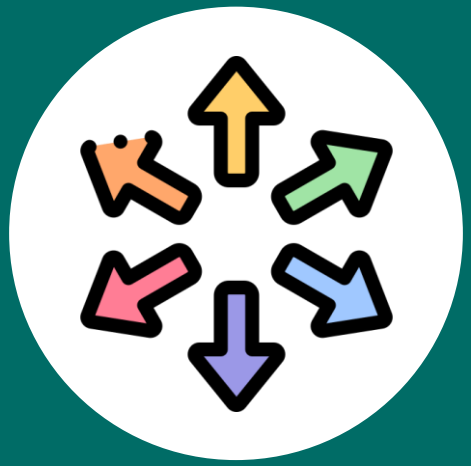
Recap

Research Question I:
What challenges and efforts are entailed with real-world FPGA reverse engineering in a black-box setting?

We reverse engineered an FPGA found in the iPhone 7

- First comprehensive FPGA reverse engineering case study on a real-world device
- Started from a bitstream and ended at a high-level understanding.

Solved the challenges we encountered either manually or with specific and custom made tooling.



Generalization

Generalization of Reverse Engineering Techniques

Research Question II:
To what extent can FPGA reverse engineering be generalized across architectures and implementations?

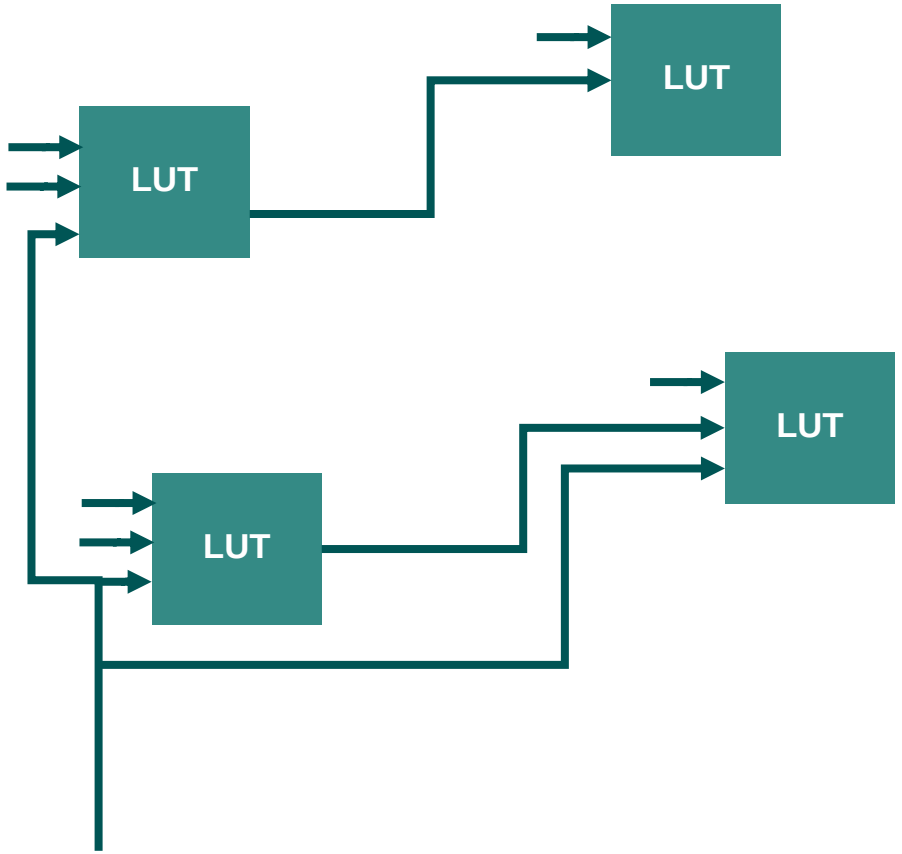
We created a general a tool box for FPGA reverse engineering.

- Consisting of multiple tools for different phases and challenges of the reverse engineering flow
- Mostly independent of architecture (Xilinx, Lattice, and extendable)
- Evaluated on set of benchmarks for both architectures and different functionalities

Netlist Pre-Processing

Resynthesis for LUT replacement

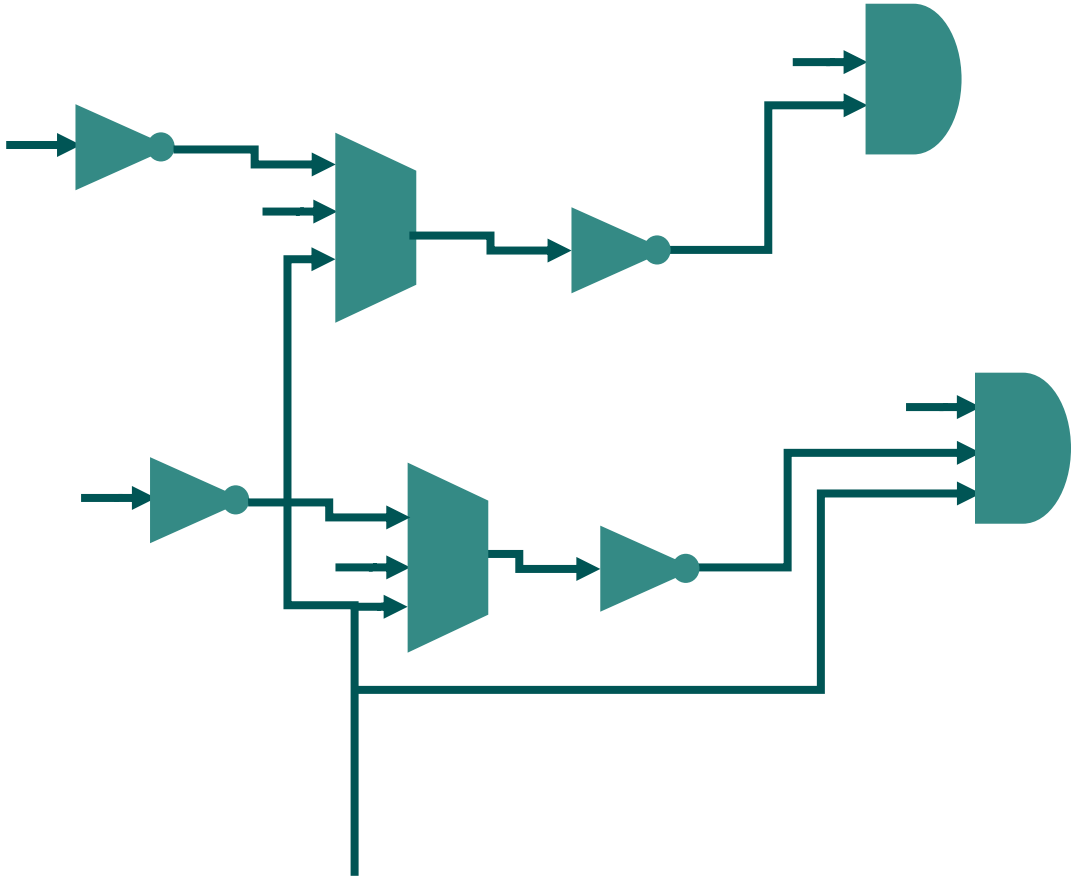
- Use existing synthesizers to map LUTs into basic gates
- Allows for reconstruction of structural information and improves readabiltiy
- Translates netlists of different architectures into a similar format



Netlist Pre-Processing

Resynthesis for LUT replacement

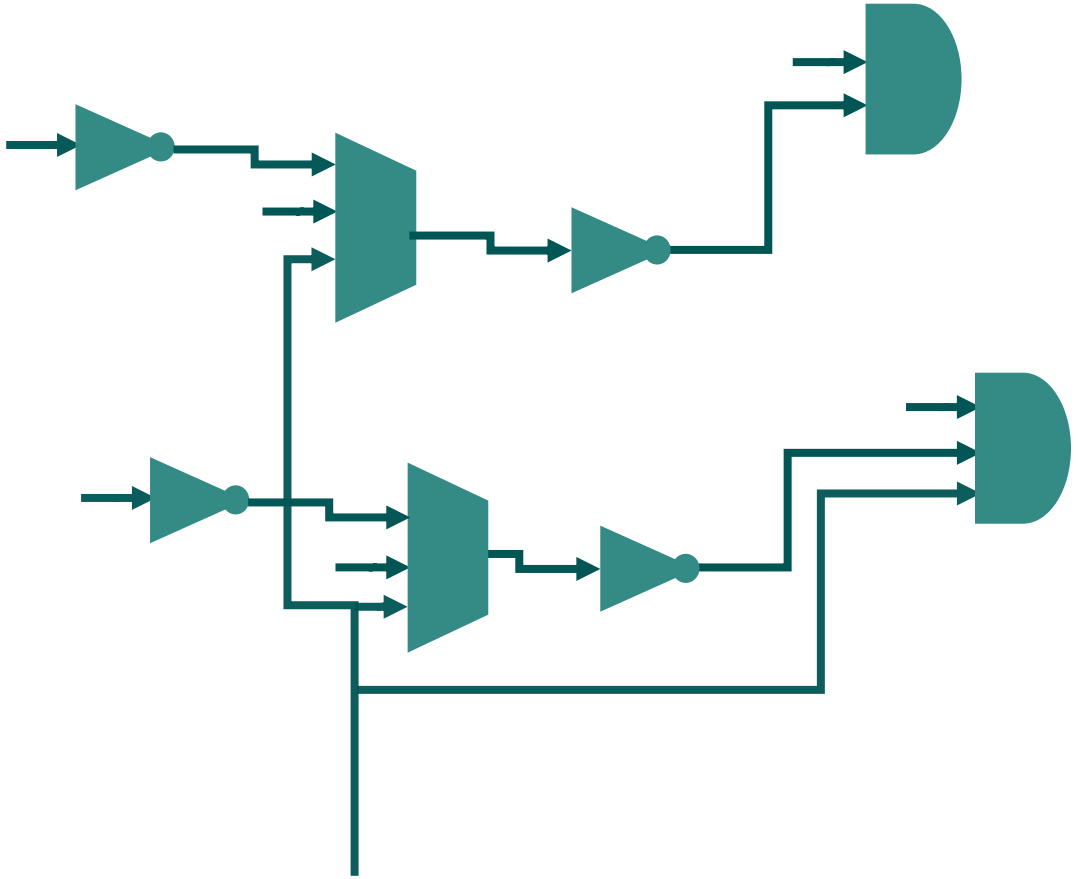
- Use existing synthesizers to map LUTs into basic gates
- Allows for reconstruction of structural information and improves readabiltiy
- Translates netlists of different architectures into a similar format



DATAPATH

New features for DANA:

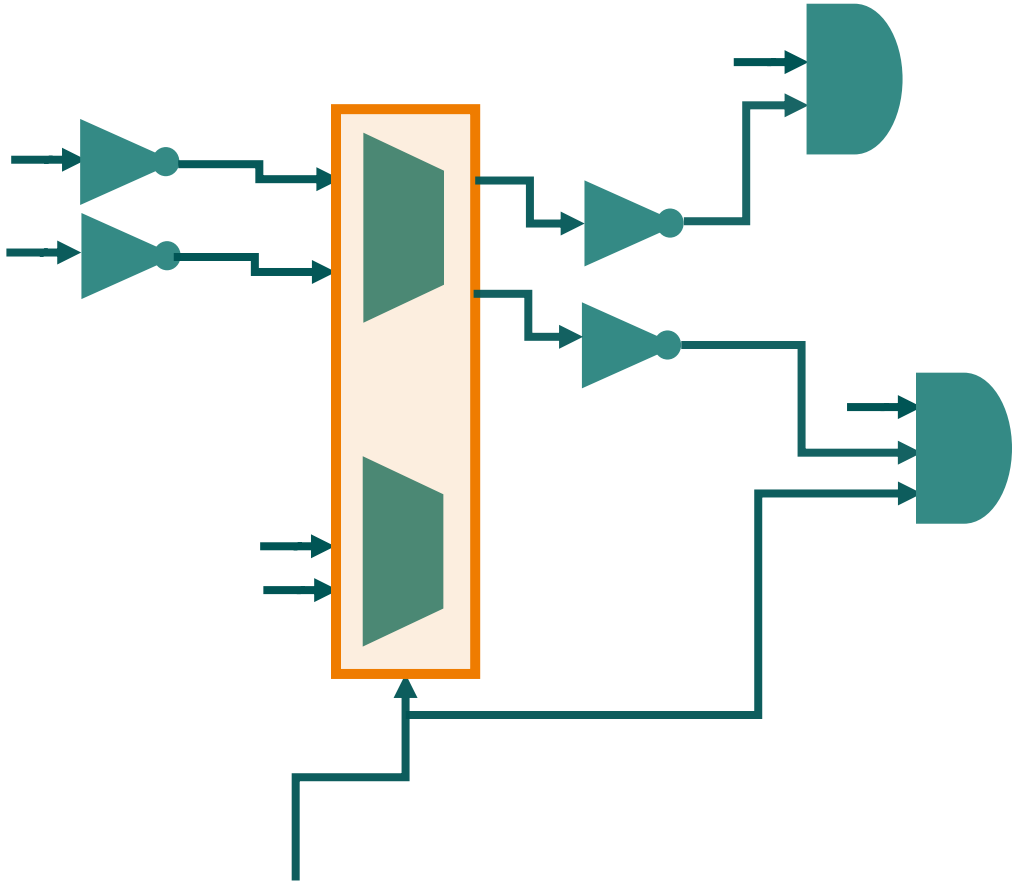
- Operate on more gates than FFs
  - Support for MUX grouping directly in DANA



DATAPATH

New features for DANA:

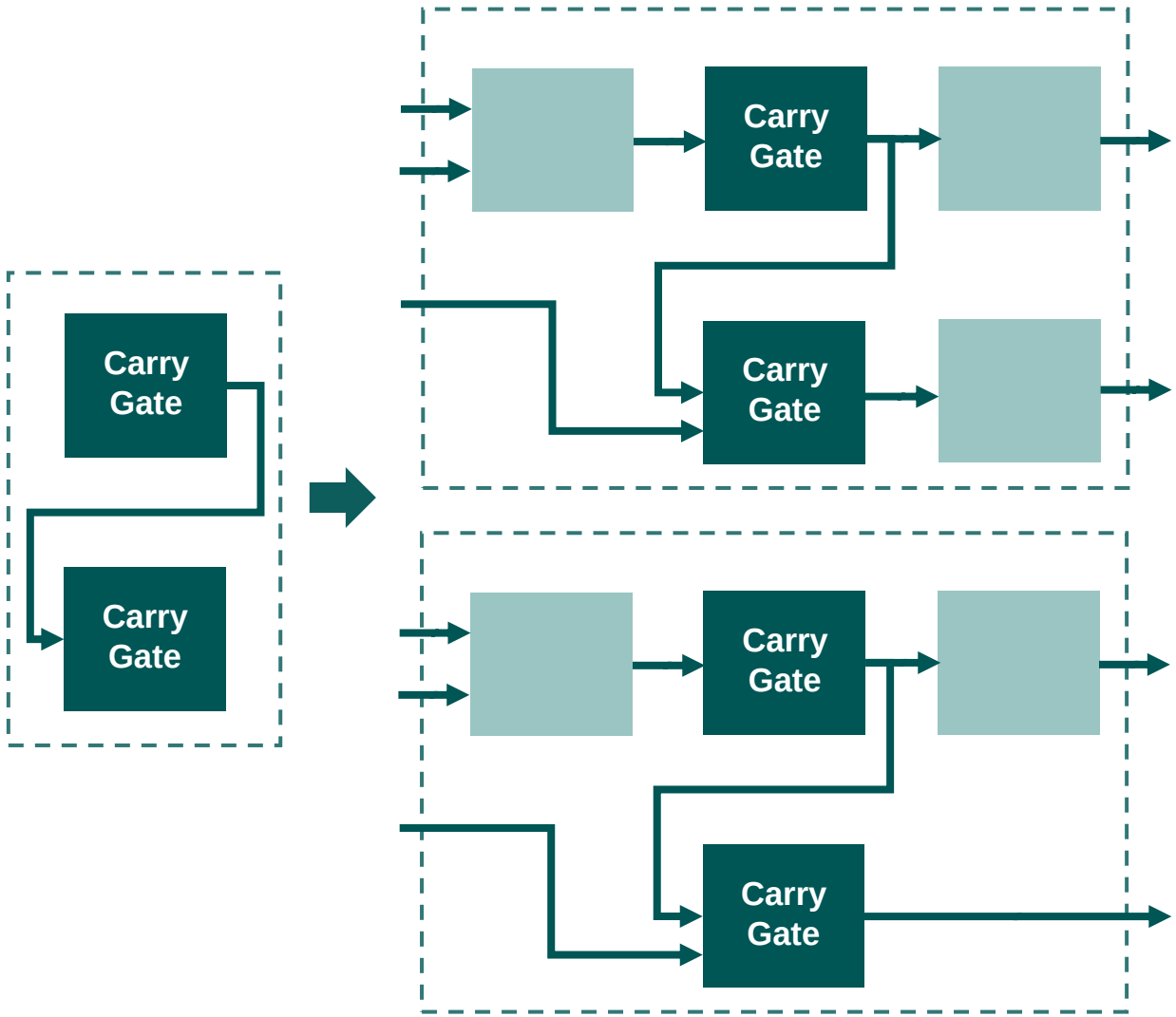
- Operate on more gates than FFs
  - Support for MUX grouping directly in DANA



Arithmetic Structures

Structural Candidate Identification:

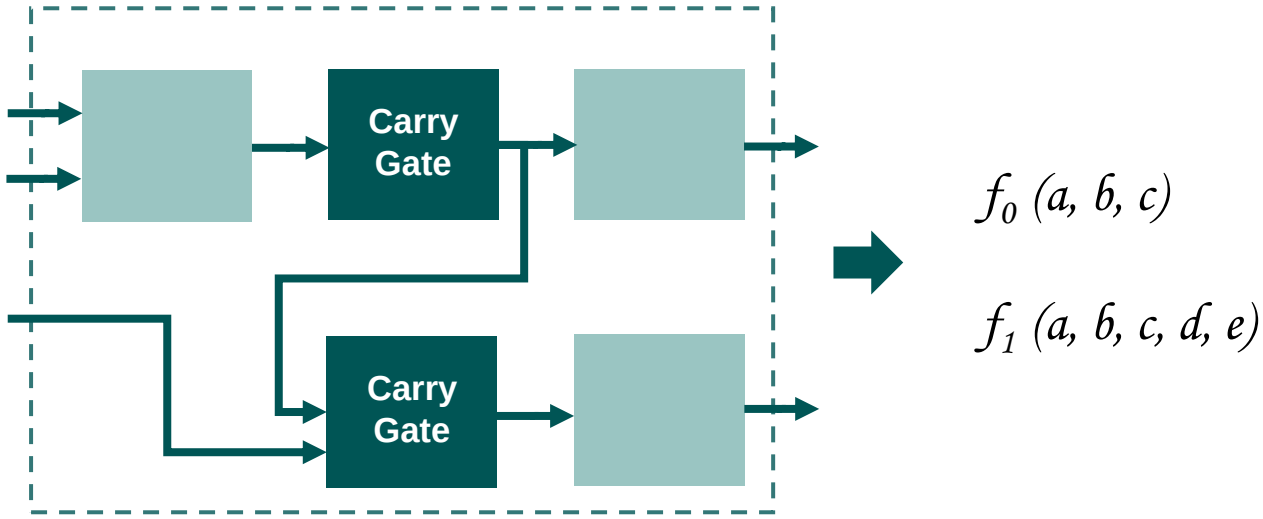
- Based on FPGA characteristics
- Starting with carry chains
- Build varying structural candidates with neighboring gates



Arithmetic Structures

Functional Candidate Identification:

- Abstract candidate into Boolean functions of output nets



Arithmetic Structures

Functional Candidate Identification:

- Abstract candidate into Boolean functions of output nets
- Identify possible control signals, input operands and output order
  - Purely based on function analysis

$f_0\ (a,\ b,\ c)$

$f_1\ (a,\ b,\ c,\ d,\ e)$

Addition-Candidate

Operands:
A: [a, d]
B: [b, e]

Output Order:
$[f_0, f_1]$

Control Signals: [c]

Addition-Candidate

Operands:
A: [a, d]
B: [c, e]

Output Order:
$[f_0, f_1]$

Control Signals: [b]

Subtraction-Candidate

Operands:
A: [a, d]
B: [b, e]

Output Order:
$[f_0, f_1]$

Control Signals: [c]

Arithmetic Structures

Functional Candidate Identification:

- Abstract candidate into Boolean functions of output nets
- Identify possible control signals, input operands and output order
  - Purely based on function analysis
- Check all generated Candidates with an SMT solver

Addition-Candidate

Operands:
A: [a, d]
B: [b, e]

Output Order:
$[f_0, f_1]$

Control Signals: [c]

Addition-Candidate

Operands:
A: [a, d]
B: [c, e]

Output Order:
$[f_0, f_1]$

Control Signals: [b]

Subtraction-Candidate

Operands:
A: [a, d]
B: [b, e]

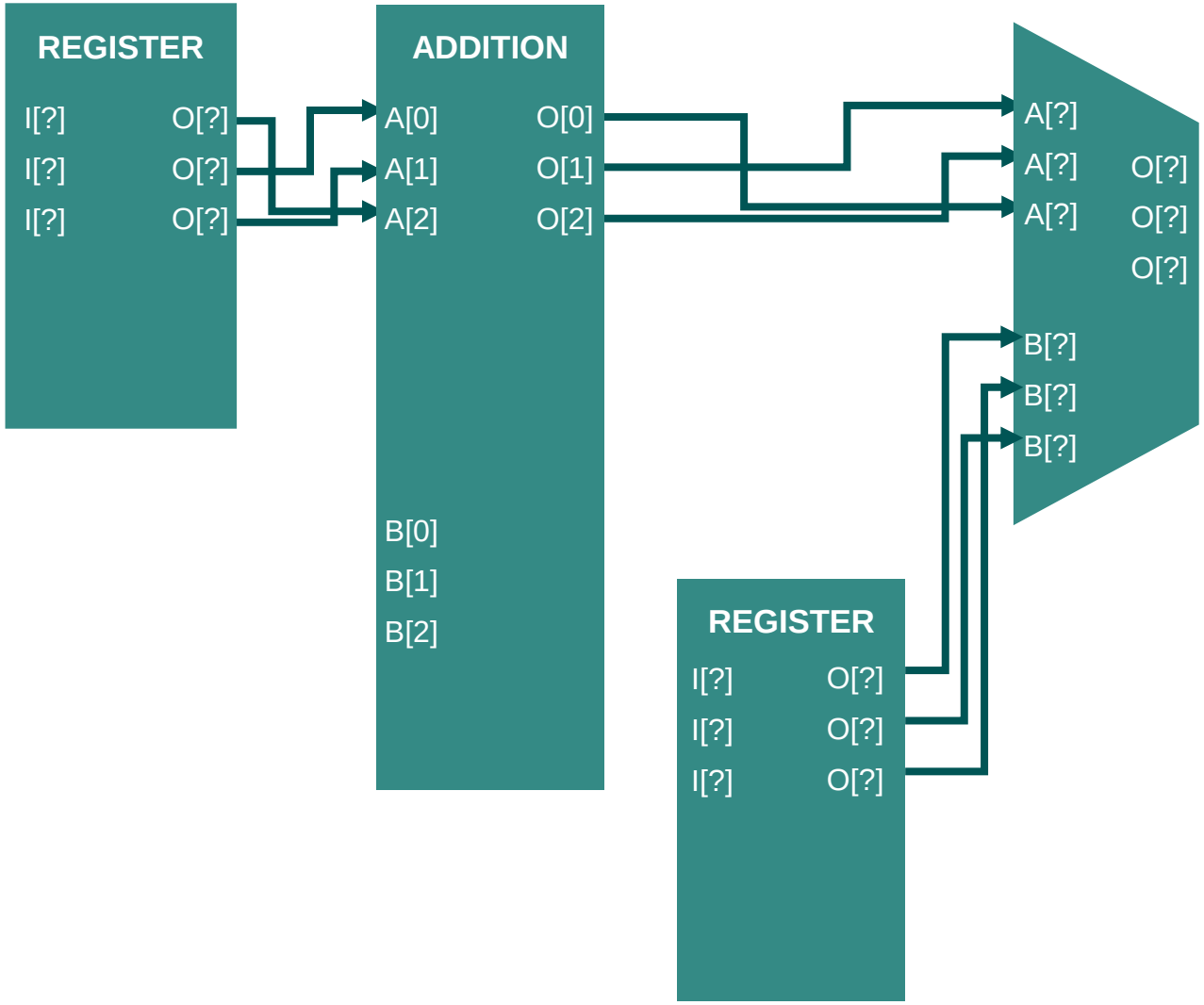
Output Order:
$[f_0, f_1]$

Control Signals: [c]

Bitorder Propagation

Working on a netlist with reconstructed word-level structures

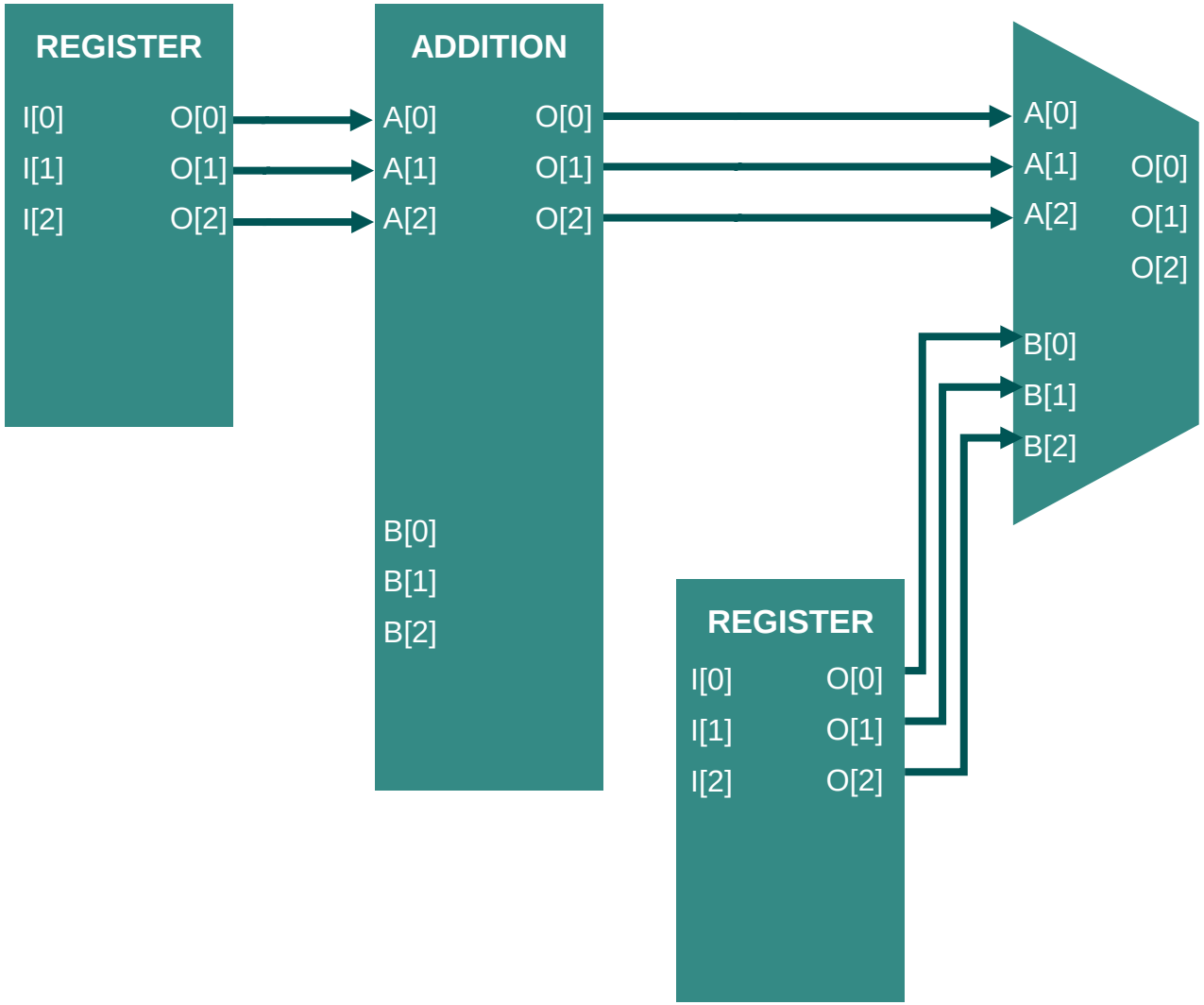
- Arithmetic structures inherently provide a bitorder of their multi-bit inputs
- Registers and Multiplexers are reconstructed with unordered IO



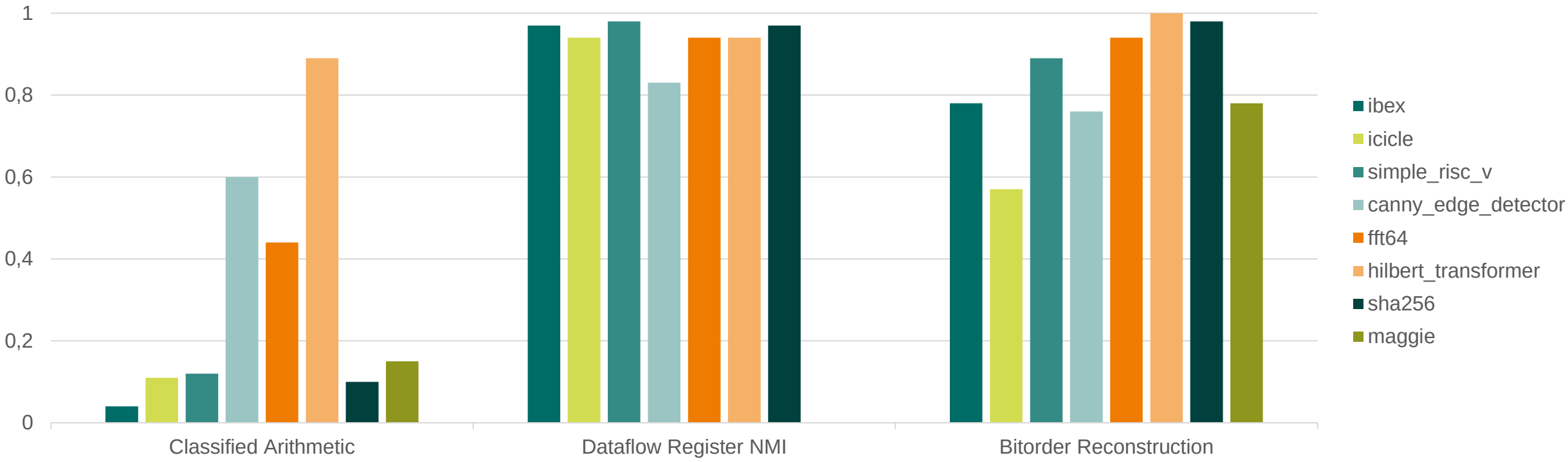
Bitorder Propagation

Working on a netlist with reconstructed word-level structures

- Arithmetic structures inherently provide a bitorder of their multi-bit inputs
- Registers and Multiplexers are reconstructed with unordered IO
- Propagate information between pingroups



Results



Large parts of the netlist can be reverse engineered fully automatically!

Threat Analysis

Threat Analysis

Research Question III:
What is the threat potential of FPGA RE?

- From previous research we have shown that netlist reverse engineering for FPGAs can be scalable
- We analyze the threat potential and compare FPGA RE to ASIC RE

Phase 1: Netlist Extraction

Differences between FPGA and ASIC Reverse Engineering

ASICs:

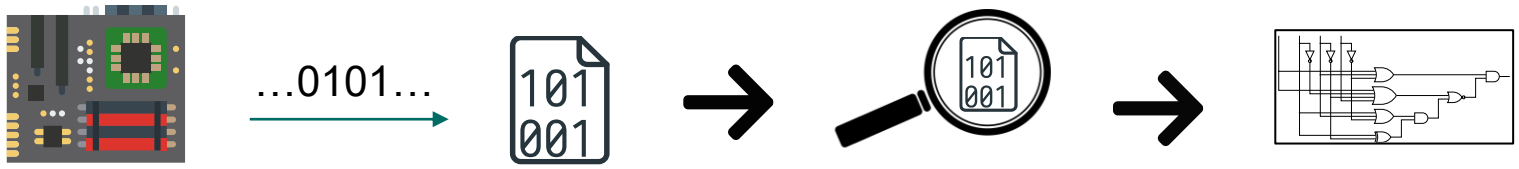
→ **Requires specialized expensive equipment and know-how**

→ **Takes weeks – months until netlist has been extracted**

Phase 1: Netlist Extraction

Differences between FPGA and ASIC Reverse Engineering

FPGAs:



Bitstream Extraction **Bitstream Conversion** **Netlist**

→ **Bitstream has to be extracted (*minutes* – *days*)**

→ **If database is available, the conversion only takes *minutes***

Open-Source Bitstream Databases

Xilinx:

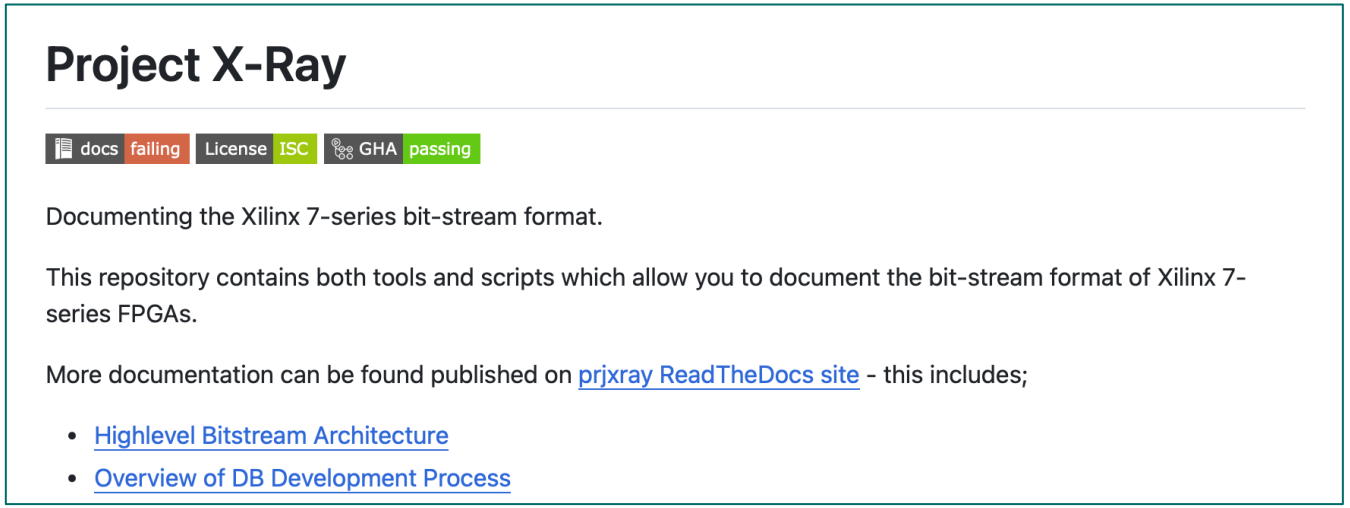
- **Project X-Ray: 7-Series**
- **Project U-Ray: UltraScale(+)**

Lattice:

- **Project Trellis: ECP5 Series**
- **Project IceStorm: ICE40 Series**

Intel:

- **Mistral: Cyclone V**



QuickLogic: Open-Source FPGAs

- QuickLogic is the first FPGA vendor to switch to 100% open-source software and hardware solution for some FPGA devices

- Databases have to be made available somewhere and can be easily accessed

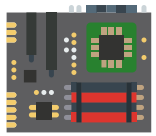
Bitstream Encryption

- **Not available for every device, or…**

- **Xilinx:**
  - **6-Series:**
    - Several side-channel attacks demonstrated
  - **7-Series:**
    - Completely broken due to protocol error (StarBleed)
  - **UltraScale(+):**
    - Partial break of encryption, if not properly configured (StarBleed-NG)
    - RSA Authentication broken (JustStart)

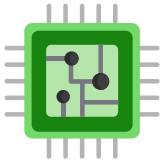
Phase 2: Netlist Analysis

Differences between FPGA and ASIC Reverse Engineering



FPGA

- **Error free netlist *can* be extracted**
  - ➢ **Enables different analysis methods that rely on exact representation (SMT, simulation, …)**
- **Not many synthesis tools and gate libraries**
  - ➢ **Reuse family specific tools**
- **Blocks, like DSPs and (B)RAMs, are already present in netlist**
- **Netlist can change over time *(dynamic reconfiguration)* → *hard to analyze***



ASIC

- **Error free netlist are *unlikely* to be extracted**
  - ➢ **Rely on *fuzzy* methods**
- **Large variety of tools and gate libraries**
  - ➢ **Tools must be adjusted more often**
- **Blocks have to be reconstructed**
- **Netlist is static**

Threat Potential and Consequences

- **FPGAs face a different kind of attacker regarding RE**
  - **ASICs:** Attackers with vast resources (nation-state actors, large companies, specialized labs…)
  - **FPGAs:** Attackers with limited resources (hobbyist, university researchers, …)
    - Combination of easier netlist extraction + readily deployable tools and algorithms

→ **More acute danger for IP implemented on FPGAs**

→ **Manipulations can be conducted**

Advanced Protection Methods

Obvious solution:

- **Develop good bitstream encryption**
- **However, bitstream encryption has failed many times in the past…**

Fallback protection mechanisms:

- **Netlist obfuscation:**
  - Has been applied in the software world for many years
  - Strong obfuscation can use FPGA specific features (like reconfigurability), but should also focus on making RE as hard as possible on gate-level reverse engineering

MAX PLANCK INSTITUTE
FOR SECURITY AND PRIVACY

Any Questions?