# P4-to-FPGA: High Performance Reconfigurable Networking

Petr Kaštovský
Netcope Technologies, a.s.
U Vodárny 2, 616 00
Brno, Czech Republic
kastovsky@netcope.com

Pavel Benáček, Viktor Puš
CESNET a.l.e.
Zikova 4, 160 00
Prague, Czech Republic
benacek,pus@cesnet.cz

## CCS Concepts

**•Hardware → Hardware description languages and compilation;**

## Keywords

P4; FPGA; High Level Synthesis

## 1. ABSTRACT

OpenFlow [1], as the most popular embodiment of Software Define Networking, provides a way to network dataplane configuration at runtime. The OpenFlow specification strictly defines a set of supported protocols and actions for further processing of incoming traffic (i.e, switches are still mostly fixed). However, modern requirements on networking hardware have a dynamic character and administrators of high-end networks want to react to new protocols, security threats, novel approaches in traffic engineering, and so on. This isn't feasible with static network hardware and leads to the need to replace the hardware more often than desired.

P4: Programming Protocol Independent Packet Processors [2], is one step ahead and evades the limitation of fixed networking devices. P4 provides a way to describe a custom packet processing chain that involves parsing, matching and assembling modified packets. This additional step of abstraction allows for even more decoupling of data plane and control plane, possibly enabling more options in virtualization of networking resources. The language is target independent and can be mapped to CPUs, FPGAs, NPUs. FPGAs are aligned with the ideas of P4 because of their structural programmability and massively parallel nature.

Our work introduces an algorithm that maps P4 to a general architecture of FPGA-based networking device. The architecture consists of three building blocks - Parser, Match+Action pipeline and Deparser. Parser transforms each incoming packet to the form of protocol headers which are passed to the Match+Action pipeline. Match+Action pipeline implements the required decision-making functionality like IP lookup or ACLs, and also modifies the protocol headers, if necessary. The Match+Action tables are populated in runtime, similar to OpenFlow. Finally, the Deparser transforms the set of possibly modified protocol headers back to the form of network packet. The whole device (and its API) is automatically generated from P4 source code and the network administrators don't need to write any HDL code. Several parameters, such as data width and pipeline depth, can be used to tune the generated circuit throughput, latency and area.

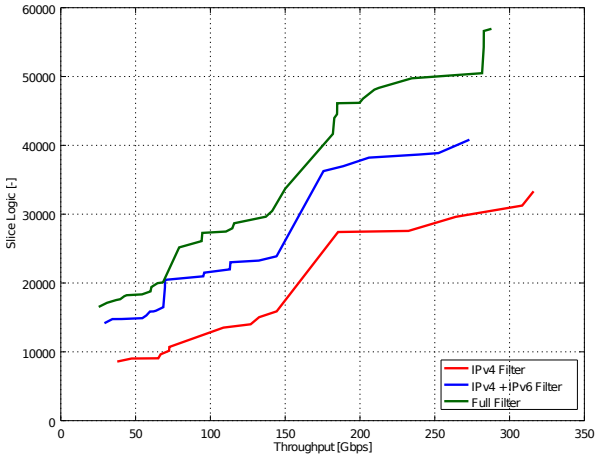


**Figure 1: FPGA resources for different P4 programs.**

Fig. 1 shows FPGA resources and throughput of generated devices for different parameters on Virtex 7 FPGA. Resources are expressed in the form of Slice Logic which is the sum of required LUTs and registers. Three use cases (or P4 programs) are shown. The *IPv4 Filter* is a simple engine where the filtering process is based on source IPv4 address. The *IPv4+IPv6 Filter* extends the previous with IPv6 protocol support. Finally, *Full Filter* further extends the previous with the ability to filter or distribute the traffic into separate VLAN/MPLS networks based on source IPv4/IPv6 address.

## 2. REFERENCES

[1] Open Networking Foundation. Open Flow. https://www.opennetworking.org/sdn-resources/openflow.

[2] P4 Language Consortium. P4. http://p4.org/.

*H2RC2016 Heterogeneous High-performance Reconfigurable Computing, Salt Lake City, UT, USA*

ACM ISBN 123-4567-24-567/08/06.

DOI: 10.475/123_4