Using P4 for converged and programmable XHaul in mobile RAN

James Yu, Nokia
Gordon Brebner, Xilinx
XHaul Overview

(a) RAN with RRH

(b) C-RAN

Fronthaul

Midhaul

Backhaul
Converged XHaul Challenges

1. Fronthaul Split
   - Core Network
   - EPC
   - L3+
   - L2/Sch
   - L1
   - Central BTS/Unit
   - RF
   - RRH

2. Split L1
   - Core Network
   - EPC
   - L3+
   - L2/Sch
   - L1'
   - Central BTS/Unit
   - RF
   - RRH + L1'

3. Split @ L2
   - Core Network
   - EPC
   - L3+
   - L2/Sch
   - L1
   - Central BTS/Unit
   - L1'
   - RF
   - RRH + L1

4. Split @ L3
   - Core Network
   - EPC
   - L3+
   - L2/Sch
   - L1
   - Central Unit
   - L2/Sch
   - L1
   - RF

5. Backhaul Split
   - Core Network
   - EPC
   - L3+
   - L2/Sch
   - L1
   - RF
   - Ethernet
   - Lower Data Rates
   - Higher Data Rates
   - eCPRI, RoE
   - NGFI, eCPRI
   - 802.1CM
   - TSN, DetNet
   - Central BTS/Unit
   - L1

This diagram illustrates the challenges and solutions for converged XHaul, focusing on the network layers and data rates involved in the process.
P4 Converged Solution

P4 Programmable PP in converged fabric

Also at NICs

PP not just protocols, but priority, classification, scheduling, and more…
## What Does P4 Solve for X-Haul?

<table>
<thead>
<tr>
<th>Issues</th>
<th>Resolution Using P4</th>
</tr>
</thead>
<tbody>
<tr>
<td>No single right solution. Multiple options with diverse requirements.</td>
<td>Single converged HW but customizable solutions using programmable P4 SW</td>
</tr>
<tr>
<td>How to inter-operate and upgrade?</td>
<td>Enhance NIC + fabric to inter-operate with SW upgrade.</td>
</tr>
<tr>
<td>Silicon follows standardization, which is far away</td>
<td>No need to wait. Evolve with standards and new protocols as they emerge.</td>
</tr>
</tbody>
</table>

- Churns → Good for using P4
Current Xilinx P4 compilation (www.xilinx.com/sdnet)

Xilinx P4 Compiler
% p4c-fpga example.p4

Xilinx PX Compiler
% sdnet example.px

Verification Environment

Top level Verilog wrapper
Verilog Engines (Encrypted)
System Verilog Testbench
High level C++ Testbench
Runtime C++ drivers

VIVADO, HLx Editions

example.p4
example.px
example.v
example.bit
Next-generation Xilinx P4-SDNet compilation
Prototyped now, scheduled for release in November 2018

Optimization of data path: Native P4 compilation
>50% less latency/resources

Open standard API: P4 Runtime
Xilinx Labs “Type 1, 2 and 3 NIC” prototype

Virtual Machines hosted on CPU
DPDK

PCle/SRIOV

Ingress datapath offload
P4

Application function acceleration
C/C++

Egress datapath offload
P4

Memory

40/50G Ethernet MAC/PHY

<table>
<thead>
<tr>
<th>Description</th>
<th>Example features</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type 1</strong> Basic Connectivity NIC</td>
<td>• Basic offloads (CHKS, LSO, RSS) • Single Root I/O Virtualization • Tunnel offloads (VXLAN, GRE)</td>
</tr>
<tr>
<td><strong>Type 2</strong> SmartNIC for Network Acceleration</td>
<td>• Encryption/Decryption (IPSec) • Virtual Switch offload (OVS) • Programmable tunnel types</td>
</tr>
<tr>
<td><strong>Type 3</strong> SmartNIC for Network + Compute Acceleration</td>
<td>• Inline Machine Learning • Inline Transcoding for Video • Database Analytics</td>
</tr>
</tbody>
</table>
Summary and Suggestions

- P4 for RAN XHaul a good fit because of churns:
  - Multiple options, requirements and standardization work
  - Need HW before standardization
  - Convergence for economy of scale and operational efficiency

- Working PoC between RRH and BBU with Xilinx P4-programmed SmartNIC

- Suggestions for P4 community:
  - Support PP capability needs identified in emerging TSN and other work
  - NICs just as important as switches, so P4 NIC model as a future architecture?