Open. Together.
ODSA Proof of Concept (POC)

Quinn Jacobson, Achronix
Jawad Nasrullah, zGlue
Jayaprakash Balachandran, Cisco
Big Picture

• **Vision:** ODSA success is an open marketplace where people offer their chiplets, and those chiplets have common interfaces to interoperate

• **Mission:** ODSA’s role is to define and prove several aspects of the concept to enable companies to be willing to make investments for future multi-party chiplet products with common interfaces

• **Actions:** ODSA’s POCs is to kick start the process
  • Tackle the chicken-and-egg problem by showing companies willing to make incrementally larger investments towards interoperable parts
  • Generate proof points to reduce resistance to making first real products (Making commercial products is beyond the scope of ODSA)
Multiple dimensions of POC effort

- Operations
  - Force information sharing at a bare die-level
  - Exposing issues of sharing sensitive business metrics
  - Validate risk and value sharing models
- Architectural
  - Validate interfaces protocols
  - Evaluate performance issues
  - Develop software programming models
- Work flow
  - Yield and cost modeling
  - Explore chiplet integration and packaging
  - Validate power distribution
  - Develop high-speed I/O solutions
POC has multiple steps

**Workflow/Software Dev POC**
- Market place microcosm
- Workflow/Operation
- Software Development

**Test chip POC**
- Chiplet interface
- Packaging

**First products**

**Market place/ecosystem**
Objectives of Workflow POC

• Pathfinding with example workflow across companies and model for market place
  • Reusable cross-company workflow
  • Faster path in future for someone to build a domain-specific multi-chiplet solution
• Bootstrapping the system by building something tangible
  • Demonstrate commitment from all partners
  • Demonstrate credibility as a group
• Inspire by building accelerator (HW+SW) with contemporary performance
  • Need to attract end users
  • Show a path to real product
Workflow POC First Thought

40G Ethernet copper
40G Ethernet optical
X8 PCIe G3 (64Gbps)
Working with legacy chips/interfaces

- Explore Operations
  - Working together
  - Sharing information
- Building Something
  - Making it real
  - Credible
- Create a microcosm for a chiplet market place
  - “Chiplet” suppliers
  - “Chiplet” integrators
  - Software suppliers
  - System builders

- Engineering challenges **NOT** addressed by first POC (will be addressed with future test chips)
  - Validate new interface for chiplets
  - Packaging and board challenges of new generation of chiplets
PChiplet (pronounced “Pichlet”)  

• Innovative way to take a first step  
• A small PCB analogy of a chiplet  
• A large PCB analogy of a substrate and package
Workflow POC Platform Architecture

- **Accelerator – A**
  - PChiplet
  - PCIe
  - ETH
  - QSFP

- **Accelerator – B**
  - PChiplet
  - PCIe
  - ETH
  - QSFP

- **Accelerator – C**
  - PChiplet
  - PCIe
  - ETH
  - QSFP

- **M.2 SSD**
  - PCIe
  - ETH

- **Host PCIe Link**
  - PCIe
  - ETH
  - QSFP

- **16**

---

Open. Together.
Supports multiple configurations

Diagram showing configurations:

- Host
- PCIe 1
- PCIe 2
- PCIe 3
- PCIe 4
- QSFP
- M.2 SSD
- Eth 1
- Eth 2
- Eth 3
- Eth 4
- PCIe 1
- PCIe 2
- PCIe 3
- PCIe 4

Configuration A, B, and C show variations in the connections and components.
## First PChiplets

<table>
<thead>
<tr>
<th>Port</th>
<th>Netronome NFP</th>
<th>Achronix FPGA</th>
<th>NXP CPU</th>
<th>SSD Jumper</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ethernet 3</td>
<td>YES</td>
<td>YES</td>
<td>NO</td>
<td></td>
</tr>
<tr>
<td>Ethernet 2</td>
<td>YES</td>
<td>YES</td>
<td>NO</td>
<td></td>
</tr>
<tr>
<td>Ethernet 1</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>PCIe 4 x4</td>
<td>NO</td>
<td>NO</td>
<td>YES x4</td>
<td></td>
</tr>
<tr>
<td>PCIe 3 x8</td>
<td>YES x8</td>
<td>NO</td>
<td>YES x4 only</td>
<td></td>
</tr>
<tr>
<td>PCIe 2 x8</td>
<td>YES x8</td>
<td>YES x8</td>
<td>YES x4 only</td>
<td></td>
</tr>
<tr>
<td>PCIe 1 x16</td>
<td>YES x8 only</td>
<td>YES x8 only</td>
<td>YES x8</td>
<td></td>
</tr>
</tbody>
</table>
Example: Smart NIC bump-in-wire
Example: Smart NIC Side-car

A

B

C

QSFP

QSFP

QSFP

QSFP

NFP

FPGA

CPU

Host

Eth 3

Eth 2

Eth 1

PCIe 4

PCIe 3

PCIe 2

PCIe 1

Eth 3

Eth 2

Eth 1

PCIe 4

PCIe 3

PCIe 2

PCIe 1

Eth 3

Eth 2

Eth 1

PCIe 4

PCIe 3

PCIe 2

PCIe 1

M.2 SSD

M.2 SSD

Data Plane

Control Plane
Example: Computational Storage

A

B

C

Host

PCIe 1

PCIe 2

PCIe 3

PCIe 4

Eth 1

Eth 2

Eth 3

QSFP

QSFP

QSFP

QSFP

FPGA

FPGA

M.2 SSD

M.2 SSD

Data Plane

Control Plane

COMPUTE

COMPUTE

Open. Together.
Workflow POC physical implementation
Workflow POC Pchiplet

NFP PChiplet
- Chiplet boundary
- Power Supply Module
- Clocks
- RESET
- BOOT FLASH
- DDR3 32 MB
- DDR3 64 MB
- I2C / JTAG
- UART
- Debug LEDs
- GPOD
- 45 x 45 mm PKG

Achronix FPGA PChiplet
- Power Supply Module
- Clocks
- RESET
- CONFIG FLASH
- DDR3 64 MB
- GPOD / I2C Links
- PCIe-1
- PCIe-2
- PCIe-3
- PCIe-4
- ETH-5
- ETH-6
- ETH-7
- 52.5 x 52.5 mm PKG

NXP PChiplet
- Power Supply Module
- Clocks
- RESET
- BOOT FLASH
- DDR4 64 MB
- QoriQ LX2150
- PCIe-1
- PCIe-2
- PCIe-3
- PCIe-4
- ETH-5
- ETH-6
- ETH-7
- 40 x 40 mm PKG
Software and Application Development

Applications

PoC Infrastructure

Host

Board

Middleware+ Custom datapaths

Driver

Communication

Customization

Dataplane

Hardware Platform

Application

Storage

Smart NIC

Security

API

API

API

Functional APIs with dataplane decomposed onto PoC components

Netronome driver extended

Communication logic

eBPF/P4

ACE Tool

GNU CC

CoreNIC

Reconfig Logic Gates

General-purpose

Netronome

Achronix

NXP

From Netronome

From Achronix

From NXP

Infrastructure to be developed

Existing Open Source

Existing Component

From Netronome

From Achronix

From NXP

Applications

From Achronix

From NXP
Need help with applications developed on POC

Applications

PoC Infrastructure

Host

Board

Software and Application Development

Application

PoC

Infrastructure

Applications

PoC

Infrastructure

Board

Host

Netronome

Driver

Custom dataplane

Communication

Customization

Dataplane

Hardware Platform

Storage

Smart NIC

Security

API

API

API

Netronome driver extended

Communication logic

eBPF/P4

ACE Tool

GNU CC

CoreNIC

Reconfig Logic Gates

General-purpose

Netronome

Achronix

NXP

From Netronome

From Achronix

From NXP

Existing Open Source

Existing Component

From Achronix

From NXP

From Netronome

Open. Together.
# POC Schedule

<table>
<thead>
<tr>
<th>Tasks</th>
<th>2019</th>
<th>2020</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mar</td>
<td>Apr</td>
</tr>
<tr>
<td>Architecture and Overall</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Component Selection, Arch</td>
<td></td>
</tr>
<tr>
<td>PoC SDV Design</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PoC Package Design</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Package Manufacturing</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software/Firmware (Bring UP and demo)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Note:** PoC Schedule outlines key tasks and milestones for the project, spanning from March 2019 to March 2020. Key areas include architecture, overall design, PoC SDV and package design, manufacturing, and software/firmware development.
Raising additional funds for POC

Made a lot of progress and partners are making substantial contributions
Need additional funding to complete POC projects
Priority today is funds to complete Workflow POC
Funding partners will get early access to PoC development platform (first hardware)
Please Join Us

We are looking for fellow travelers for all areas

• Specification Write Ups
• System Netlist Verification
• Board Design
• Software Development
• Application Porting
• Board Bring Up
• Package Design
• System Level Test Development
• Sponsor Board Manufacturing
• Sponsor Package Manufacturing

Reach out to jawad@zglue.com
Open. Together.

OCP Regional Summit
26–27, September, 2019