From Cores to Chiplets: PULP's Adventure in Open-Source HPC

Gianna Paulin  pauling@iis.ee.ethz.ch
and the PULP team

PULP Platform
Open Source Hardware, the way it should be!
Cost/Yield increases with more advanced feature nodes

Increasing the monolithic die area is getting more and more costly for newer technology nodes

Xilinx: First 2.5D FPGA Chiplet Design in 2011

eSilicon: Virtex-7-like estimation for 40nm process

• Baseline of 24mm x 24mm monolithic chiplet: 25% yield
• Splitting partitioning into four 24mm x 6mm dies: 70% yield
• Even with the additional cost of the interposer the cost saving >50%

For better manufacturing yield (to save cost), a very large SoC has been split into 4 smaller chips.
Chiplet Design and Packaging Technologies

**Split Logic**
- Logic
- Logic1
- Logic2
- Logic3
- Split
- SoC
- Chiplet Designs

**Partition**
- I/O
- I/O
- Partition
- Logic
- I/O
- SoC
- Chiplet Designs

**2D / Stnd. Package**
- Organic substrate
- DRAM
- Die 1
- Die 2
- Substrate

**2.5D / Adv. Package**
- Interposer
- HBM
- Die 1
- Die 2
- Interposer Die
- Substrate

**3D / Die Stacking**
- Hybrid Bonding
- Die 2
- Die 1
- Substrate

---

John H. Lau, Unimicron Technology Corporation, “Chiplet Design and Heterogeneous Integration Packaging”, SWISS IEEE/EPS and SSCS Lecture
Our latest design Occamy: 0.75 TFLOP/s, 400+ cores

Dual Chiplet System Occamy:
• 216+1 RISC-V Cores per chiplet
• 0.75 TFLOP/s entire system
• GF12LPP
• Area: 73mm$^2$

2x 16GByte HBM2e DRAMs Micron

2.5D Integration

Silicon Interposer Hedwig:
• Technology: 65nm, passive (only BEOL)
• Area: 26.3mm x 23.05mm

Carrier PCB:
• RO4350B (Low-CTE, high stability)
• 52.5mm x 45mm
How did we get here?

Concept architecture presented at Hotchips 2020 conference [1]

- (Quad-) Chiplet-based architecture
- AI/HPC focused
- Essential components have been manufactured in GF22
- Measured for energy-efficiency
- Extrapolation on larger AI workloads (full training and inference steps)

Not All Programs Are Created Equal

• Processors can do two kinds of useful work:

  **Decide** (jump to different program part)
  - Modulate flow of *instructions*
  - **Smarts:**
    - Don’t work too much
    - Be clever about the battles you pick (e.g., search in a database)
  - Lots of decisions
    Little number crunching

  **Compute** (plough through numbers)
  - Modulate flow of *data*
  - **Diligence:**
    - Don’t think too much
    - Just plough through the data (e.g., machine learning)
  - Few decisions
    Lots of number crunching

• Many of today’s challenges are of the *diligence* kind:
  - Tons of data, algorithm ploughs through, few decisions done based on the computed values
  - “Data-Oblivious Algorithms” (ML, or better DNNs are so!)
  - Large data footprint + sparsity
Snitch – a Tiny 32b Integer RISC-V Core

- **Simplest core: around 20KGE**
  - Speed via simplicity (1GHz+)
  - L0 Icache/buffer for low energy fetch
  - Shared L1 for instruction reuse (SPMD)

- **Extensible** → “Accelerator” port
  - Minimal baseline ISA (RISC-V)
  - Performance through ISA extensions (via accelerator port)

- **Latency-tolerant** → Scoreboard
  - Tracks instruction dependencies
  - Much simpler than OOO support!
Custom ISA extensions

**FREP: Remove control flow overhead [2]**

- Programmable **micro-loop buffer**
- **Sequencer** steps through the buffer, independently of the FPU
- **Integer core free to operate in parallel:**
  **Pseudo-dual issue**

```plaintext
mv r0, zero
loop:
  addi r0, 1
  fmadd r2, ssr0, ssr1
  bne r0, r1, loop
  frep r1, 1
  fmaddd r2, ssr0, ssr1
```

**SSRs: Turn Reg R/W into Mem LD/ST [3]**

- **Address generation** hardware to Register file
- **SSRs ≠ memory operands**
  - Perfect **prefetching**, latency-tolerant

```plaintext
loop:
  fld r0, %[a]
  fld r1, %[b]
  fmaddd r2, r0, r1
  scfg 0, %[a], ldA
  scfg 1, %[b], ldB
  fmaddd r2, ssr0, ssr1
```


Architectural Innovation in Occamy

• **FPU** with **SIMD Mini-float** (ML training, Transformers) and **expanding SDOTP Unit** [4]

<table>
<thead>
<tr>
<th>Format</th>
<th>11</th>
<th>52</th>
</tr>
</thead>
<tbody>
<tr>
<td>fp64</td>
<td></td>
<td></td>
</tr>
<tr>
<td>fp32</td>
<td>8</td>
<td>23</td>
</tr>
<tr>
<td>fp16alt</td>
<td>8</td>
<td>7</td>
</tr>
<tr>
<td>fp16</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>fp8</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>fp8alt</td>
<td>4</td>
<td>3</td>
</tr>
</tbody>
</table>


• **Sparsity support** [5] (Stencils, Sparse Tensors)
  • Extend 2 out of 3 SSRs to **ISSRs** and add index comparison unit
  • Forward result indices to 3rd SSR

**Transprecision FPU by Luca B. on Tuesday at 2pm**

• **Atomsics and fast interrupts** (synchro & offload accel.)
• **I-Cache** hierarchy

**Sparsity Extensions by Paul on Tuesday at 9am**
Occamy Cluster – 8 MGE
Occamy Cluster – 8 MGE

8 Snitch compute cores
- Single-stage, small Integer control core

9th Core: DMA
- 512 bit data interface
- HW support to autonomously copy 2D shapes
- Higher-dimensionality can be handled by SW

128 kB TCDM
- Scratchpad for predictable memory accesses
- 32 Banks
Efficiently Move Data

- 64-bit AXI DMA
- Operates on **wide 512-bit data-bus**
- **Hardware support** to autonomously copy 2D shapes
- Higher-dimensionality can be handled by SW
- Intrinsics/library for easy programming
- Exploiting cluster local memory

Data Movers by Thomas on Tuesday at 3.30pm

64GB/s @1GHz per cluster
Occamy Cluster – 8 MGE

8 Snitch compute cores
- Single-stage, small Integer control core

9th Core: DMA
- 512 bit data interface
- HW support to autonomously copy 2D shapes
- Higher-dimensionality can be handled by SW

128 kB TCDM
- Scratchpad for predictable memory accesses
- 32 Banks

Custom ISA extensions
- Xfrep, Xssr
- New: Xissr sparsity support

GEMM $\geq 80\%$ FPU util.
Conv2d $\geq 75\%$ PFU util.
Stencils $\leq 60\%$ FPU util.
Sparse Tensors $\leq 50\%$ FPU util.
Occamy Cluster – 8 MGE
Four Snitch Clusters form a Group

4 Clusters per Group
- Single-stage, small Integer control core

2 AXI Busses
- 64-bit narrow interface: config
- 512-bit wide interface: DMA

Constant Cache
- D/I-Cache hierarchy

Translation Lookaside Buffers (TLBs)
- Virtual Addressing
- 8 PTEs for each (narrow and wide)
- enables:
  - Core access range extension
  - Per-group page remapping
  - Per-group access control
Four Snitch Clusters form a Group

**6 Groups:** 216 cores/die
- 4 cluster / group:
  - 8 compute cores / cluster
  - 1 DMA core / cluster
  - 512bit Constant Cache

**Linux-capable manager core** CVA6

**2 AXI Busses**
- 64-bit narrow interface: config
- 512-bit wide interface: DMA

**Peripherals**
- SPI, JTAG, I2C, UART, serial link

**8-channel HBM2e (16GB)**

**Serial DDR D2D link**
- Source synchronous
- Technology-independent
- Moderate speed: ≤ 125MHz
We designed our own custom Die-to-Die link

- Availability of HBI IP unfortunately not aligned with project timeline
- **Custom solution:**
  - **Source synchronous**
  - Bunch-of-wires (BOW) style
  - **DDR**
  - technology-independent
    - Standard digital pads
    - Moderate speed: ≤ 125MHz
    - Front-end + controller usable for faster PHYs

**Network Layer:**
- Full AXI4 interface
- AXI4 to AXI stream converter

**Data Link Layer**
- Credit-based flow control
- RX synchronisation

**Channel Allocator**
- Chops and reshuffles payload
- **Fault tolerance** mechanisms

**Physical Layer**
- Source-synchronous DDR sampling
- Scalable number of channels
  - (38 x 8-bit full-duplex DDR channels in Occamy)

https://github.com/pulp-platform/serial_link
Main Compute architecture is open-source !!!

The main compute architecture is being developed fully open-source !!!

HBM, DFG, FLL, and any proprietary components are in a separate private repository on our internal Gitlab

github.com/pulp-platform/serial_link

github.com/pulp-platform/snitch
Main Compute architecture is open-source !!!

**Chiplet Occamy:**
- Technology: GF12LP+
- 1 GHz
- Area: 73mm$^2$
- >1 billion transistor

<table>
<thead>
<tr>
<th>Precision</th>
<th>Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>FP64</td>
<td>384 GFLOp/s</td>
</tr>
<tr>
<td>FP32</td>
<td>768 GFLOp/s</td>
</tr>
<tr>
<td>FP16</td>
<td>1,536 TFLOp/s</td>
</tr>
<tr>
<td>FP8</td>
<td>3.072 TFLOp/s</td>
</tr>
</tbody>
</table>

Taped out: 1$^{st}$ of July 2022
Balancing Bandwidth and Compute

**Problem:** HBM Accesses are not ideal in terms of
- Access energy
- Congestion
- High latency

Instead reuse data on lower levels of the memory hierarchy
- Between **clusters**
- Across **groups**

Smartly distribute workload
- **Clusters:** *DORY* [5] / *Deeploy* framework for deployment / tiling strategy
- **Chiplets:** E.g. Layer pipelining

---


**Cluster-to-Group**
- 6x 256 GB/s
- 1.536 TB/s

**Group-to-Group**
- 6x64 GB/s
- = 384 GB/s

**HBM PHY**
- 512 GB/s

**HBM DRAM**
- <410 GB/s

**19.5 GB/s**

**Die-to-Die Serial Link**

**Group Crossbar**

**Simplified System Crossbar**

**DMA**

**NAS/Deeploy by Alessio, Mortiz on Tuesday at 11.30am**
Programming Model

• Multiple layers of abstraction:
  • Hand-tuned assembly
  • LLVM intrinsics
  • MLIR support for Snitch
  • High-level frameworks:
    • DaCE: spcl.inf.ethz.ch/Research/DAPP/
    • Dory / Deeploy: deployment and tiling of NNs
• Bare-metal runtime
• Basic OpenMP runtime
• HERO toolchain
Occamy mapped onto **2x VCU128 (with HBM) + 1x VCU1525**

- 1x CVA6
- 2-4x Snitch clusters

Attaching a **Xilinx PCIe controller & PHY**

- x86 Linux host supported
- RISC-V Linux host *(Monte Cimone)* in progress

Supporting **hybrid usage** (High SW stack re-usability)

- Boot directly on standalone CVA6
- Do not boot and let the Host control the cluster
Our Silicon Interposer Hedwig (65nm, passive, GF)

- **Die-to-Die**: ~600 wires
- 0.25 Gb/s per pin
- Channels are routed together as a bus/bundle, with a **matched length constraint (8.8mm)** in “bundels” of 3-4 channels

Taped out: 15th of October 2022
Our Silicon Interposer Hedwig (65nm, passive, GF)

- HBM: ~1700 wires
  - 3.2 Gb/s per pin
  - Nearly 100% track utilization
  - Ground shielding planes
  - HBM wire length is 4.9mm
Our Silicon Interposer Hedwig (65nm, passive, GF)

- **Interlocked** die arrangement
  - Prevent bending, increase stability
- **Compact** die arrangement
  - No *dummy dies* or *stitching* needed
- **Fairly low I/O pin count** due to no high-bandwidth periphery
  - Off-package connectivity: ~200 wires
  - Array of **40 x 35 (-1) C4s** (total of 1’399 C4 bumps)
    - Diameter: 400µm, Pitch: 650µm

Taped out: 15th of October 2022
25% of the area is HBM CTRL

39% of the area is COMPUTE

11% of the area is D2D

<23% of the area is CROSSBARS
Carrier PCB brings mainly “fan-out” for PCB mounting

**Carrier PCB (52.5 x 45mm)**

- Material Selection: RO4350B
  - low Coefficient of Thermal Expansion (CTE)
- High stability
- Decoupling caps
- Custom ZIF socket design
Finished Chiplet Tapeout in less than 15 months

• Initial discussions 20\textsuperscript{th} of October 2020
• Started on 20\textsuperscript{th} of April 2021
• Taped out Chiplet on 1\textsuperscript{st} of July 2022
• Taped out Interposer on 15\textsuperscript{th} of October 2022
• Currently being assembled

There is much more to come in Q3-2023 ...
There is much more to come in Q3-2023...

Peak System perf. @1GHz:
FP64: 768 GFLOp/s
FP32: 1.536 TFLOp/s
FP16: 3.072 TFLOp/s
FP8: 6.144 TFLOp/s