Datacenter FPGA accelerators await our apps

- MS Catapult, Amazon AWS F1, Alibaba, Baidu, NimbleX
- Massively parallel, specialized, connected, versatile
- High throughput, low latency, energy efficient

But two hard problems

- Software: Porting & maintaining workload as accelerator
- Hardware: Compose 100s of cores, 100G NICs, many DRAM/DRAM channels, with easy timing closure

Mission: GRVI Phalanx FPGA accelerator kit

- GRVI: FPGA-efficient RISC-V processing element cores
- Phalanx: array of clusters of PEs, SRAMs, accelerators
- Hoplite NoC: FPGA-optimal directional 2D torus soft NoC
- Local shared memory, global message passing, PGAS

Software-first, software-mostly accelerators

- Run your C/OpenCL kernels on 100s of soft processors
- Add custom function units/cores/services to suit
- More 10 sec compiles, fewer 5 hour synth/place/route
- Complements high level synthesis & OpenCL

2017: V1: 1680 core GRVI Phalanx in a VU9P

2019: V1 shortcomings

- 32b pointers poor for AWS F1 / big data / OpenCL kernels
- 32b accesses waste half 64b UltraRAM cluster bandwidth
- In-order arch stalls on loads = 5 cycles in an 8-PE cluster
- Packed, congested, insuff. pipelining = low freq (300 MHz)
- DRAM (10s GB/s) uncompetitive vs. GPUs (100s GB/s)

2019: game changer: FPGAs ↔ HBM2 DRAM

- Xilinx UltraScale+: VU3xP and Intel Stratix 10 MX families
- Xilinx: two HBM2 stacks; 32 AXI-HBM bridge/controllers
- AXI-μMCS switch: any AXI port can access any controller
- Reads/writes up 32 x 256b x 450 MHz = 460 GB/s

Let's make it easy to use this bandwidth

- TB/s to BRAMs/ULTRAMRs, 100s GB/s to HBM2 DRAM

V2: redesign GRVI Phalanx for HBM FPGAs

- New latency tolerant RV64I PEs
- New 64b cluster interconnect, 64b UltraRAM banks
- New 32B/cycle deep pipeline NoC-AXI RDMA bridges
- Add PCIe XDMA master (1 AXI-HBM channel)
- Add more many NoC ring columns
- Add Fmax-floorplanning, pipelining, FIFOs: target 400 MHz
- Add SDAccel-for-RTL shell if, OpenCL kernel runtime
- 15 cols x 256b x 400 MHz – peak 192+192 GB/s+R+W?
- T: work-in-progress / pending / current plan

FPGA soft processor area and energy efficiency

- Simpler CPUs -- more CPUs -- more memory parallelism
- Deconstruct PEs into minimal core plus cluster-shared concurrent FUs: shifts, mul, custom FUs, memory ports (!)

2GRVI -- a simple, latency tolerant RV64I PE

- 400 LUTs (sans shared barrel shifter), up to 550 MHz
- Register scoreboard: only stall on use of a busy register
- Out of order retirement; concurrent execution
- Call a save req reloads, block copies: now 1 load/cycle
- 2 stage: [DC] EX -- 3 stage: IF [DC] EX -- 4 stage: IF [IF] DC] EX\WB
- 4 stage (superpipelining) has L=2 ALU – CPI ↑ 25%
- Plan: further tolerate latency with two hardware threads

Cluster: 0-8 PEs, 128 KB RAM, accel.'s, router

- Compose cores & accelerator(s), & send/receive 32 byte messages via multiported banked cluster shared RAM

GRVI Phalanx: W.I.P. Towards Kilocore RISC-V® FPGA Accelerators with HBM2 DRAM

- 1776 RV32I / 1332 RV64I cores, 28 MB SRAM, 30 HBM2 DRAM Channels, PCIe, on Xilinx UltraScale+ VU37P / Alveo U280

Jan Gray | Gray Research LLC | Bellevue, WA | jan@fpga.org | http://fpga.org

Cluster, load-interval

Two HW threads

Out-of-order retire - typical but optional

Clusters, load-to-use

5 cycles

6 cycles / 3 thread-cycles†

Cluster, 1 RAM BW

4.8 GB/s (300 MHz)

12.8 GB/s (400† MHz)

GRVI PE

GRVI PE

<table>
<thead>
<tr>
<th>Year</th>
<th>GRVI Target</th>
<th>UltraScale+</th>
</tr>
</thead>
<tbody>
<tr>
<td>2019 Q4</td>
<td>20 nm UltraScale</td>
<td>16 nm UltraScale+</td>
</tr>
<tr>
<td>2019 Q4</td>
<td>Verilog</td>
<td>System Verilog</td>
</tr>
<tr>
<td>2019 Q4</td>
<td>ISA RV32i + mufl/isc</td>
<td>RV4 idi + mufl/isc</td>
</tr>
<tr>
<td>2019 Q4</td>
<td>Area 320-6 LUTs</td>
<td>400-6-LUTs</td>
</tr>
<tr>
<td>2019 Q4</td>
<td>Times/ongoing</td>
<td>400 / 300 MHz</td>
</tr>
<tr>
<td>2019 Q4</td>
<td>Pipeline stages</td>
<td>2 / 3</td>
</tr>
<tr>
<td>2019 Q4</td>
<td>Out-of-order retire</td>
<td>typical but optional</td>
</tr>
<tr>
<td>2019 Q4</td>
<td>Two HW threads</td>
<td>optional (= CLUS LUTs)</td>
</tr>
<tr>
<td>2019 Q4</td>
<td>Cluster, load interval</td>
<td>5 cycles</td>
</tr>
<tr>
<td>2019 Q4</td>
<td>Cluster, load-to-use</td>
<td>5 cycles</td>
</tr>
<tr>
<td>2019 Q4</td>
<td>Cluster, 1 RAM BW</td>
<td>4.8 GB/s (300 MHz)</td>
</tr>
<tr>
<td>2019 Q4</td>
<td>12.8 GB/s (400† MHz)</td>
<td></td>
</tr>
</tbody>
</table>

Kilocore GRVI and 2GRVI HBM2 Phalanxes, Now Running in a Xilinx VU37P-ES1 in an Alveo U280-ES1

- 1776 GRVI @ 300 MHz: first kilocore RV32I with HBM2
- “SOC Woods: estimated power of 100W / 64LUT/26 cores 256KRAM+S-HBM=4.8GB/s (128B AXI-HBM bus)
- 1332 2GRVI @ 1 Mhz: first kilocore RV64I with HBM2

PE--cluster RAM==>NoC--AXI==>HBM design

PEs send write / read-burst requests to a NoC-AXI bridge

- 32B writes/32B reads, split trans, deeply pipelined

- Bridge queues, issues R/W to an AXI-HBM

- Bridge queues, sends read response messages over NoC

- Per-PE and per-cluster R/W order preserved

- Requests/responses on Y-rings -> in-order delivery

- Never queue in Y-rings -> request ingress flow control

NoC-AXI RDMA bridge future research

- Bandwidth studies
- Small LCLs at NoC-AXI RDMA bridges?
- Software defined access reordering – SW sets AXI txn IDs
- “Computational HBM” – compute offload at AXI bridges
- Scatter/gather, add-to-memory, block zero, copy, checksum, reduce, select, reexecute, sort, decompless, ...

Xilinx VU3xP HBM2 first impressions

- The 32 hard 256B AXI-HBM bridges are easy to design to

- Software defined access reordering – SW sets AXI txn IDs

- Easier DRAM interface timing closure†

- Max bandwidth: longer bursts & avoid switch; use NoC?

- Nontrivial to access and transport all that bandwidth

Democratizing HBM memory systems

- HBM technology now accessible to any engineer
- HW: VU3xP in Alveo U280 / USO; cloud instances?
- SW: OpenCL via Xilinx SDAccel; 1H20 2GRVI Phalanx

Images: Xilinx.com

– stay tuned