AHA Retreat 2021
Day 2 Introduction

Priyanka Raina
praina@stanford.edu

August 26, 2021
Motivation

With the slowdown of technology scaling, **hardware specialization** is the most promising technique for continued performance scaling of applications.

Modern SoCs have dozens of domain-specific accelerators:
- Machine Learning
- Image Processing
- Video Coding
- Cryptography
- Depth Processing
Impediments to Adopting Hardware Specialization at Scale

- Lack of a **structured approach for evolving the software stack** as the underlying hardware becomes more specialized
- Prohibitive **NRE costs** associated with verification and design of such systems
- No general **methodology for specializing hardware to domains**, rather than a few benchmarks
Agile Approach to Accelerator Design

Accelerator must be **configurable**
- So we can map new or modified applications to it (although with lower efficiency)

Hardware and compiler must **evolve together**
- Any change in hardware must propagate to compiler automatically
CGRAs as Accelerator Templates

- Think of a CGRA as an accelerator template
- Is programmable to accommodate application evolution
- Can specialize for increasing efficiency
- Exploits parallelism and locality by design
CGRAs as Accelerator Templates

- By **tuning the amount of configurability** in CGRA PEs, MEMs and the interconnect, we can create more specialized (closer to ASICs) or more general-purpose accelerators (closer to FPGAs)

More importantly, thinking of accelerators as specialized CGRAs provides a **standard accelerator template for a compiler to target**!
Automatically Generate HW and Compiler Collateral from a Single Source of Truth

Formal Specifications for Accelerator Components
- PEak Program (PE Spec)
- Lake Program (MEM Spec)
- Canal Program (Interconnect Spec)

PE Compiler
- PE HW

Lake Compiler
- MEM HW

Canal Compiler
- Interconnect HW

CGRA Generator
- CGRA Hardware (Verilog)

Physical Design Flow
- CGRA Chip

Application
- Halide Compiler
- CoreIR Dataflow Graph
- PE and MEM Mapper
- Mapped CoreIR Graph
- Place & Route Engine
- CGRA Bitstream

Rewrite Rules
- PE
- MEM

Routing Graph
<table>
<thead>
<tr>
<th>Time</th>
<th>Session</th>
<th>Speaker</th>
</tr>
</thead>
<tbody>
<tr>
<td>9:10 – 9:40</td>
<td>Amber SoC: Architecture, Design, and Evaluation</td>
<td>Alex Carsello</td>
</tr>
<tr>
<td>9:40 – 10:10</td>
<td>Automated Design Space Exploration of CGRA Processing Element Architectures using Frequent Subgraph Analysis</td>
<td>Jack Melchert</td>
</tr>
<tr>
<td>10:10 – 10:30</td>
<td>Break</td>
<td></td>
</tr>
<tr>
<td>10:30 – 10:45</td>
<td>Virtualization of CGRA Based Accelerators</td>
<td>Priyanka Raina</td>
</tr>
<tr>
<td>10:45 – 11:00</td>
<td>Unlocking Scalable Quick Error Detection with Design for Verification</td>
<td>Clark Barrett</td>
</tr>
<tr>
<td>11:00 – 11:45</td>
<td>Breakout Session: How to Democratize Hardware/Software Design</td>
<td></td>
</tr>
<tr>
<td>11:45 – 12:10</td>
<td>Breakout Session Reports</td>
<td></td>
</tr>
<tr>
<td>12:10 – 1:30</td>
<td>Lunch</td>
<td></td>
</tr>
</tbody>
</table>
# AHA Retreat Day 2 Schedule

<table>
<thead>
<tr>
<th>Time</th>
<th>Session</th>
</tr>
</thead>
<tbody>
<tr>
<td>12:10 – 1:30</td>
<td>Lunch</td>
</tr>
</tbody>
</table>
| 1:30 - 2:00  | Smart Components -- Using Session Types to Avoid Late Stage Design Bugs  
                Lenny Truong                                                        |
| 2:00 - 2:30  | Accelerating Sparse Applications                                       
                Fred Kjolstad                                                      |
| 2:30 – 3:00  | Break                                                                  |
| 3:00 – 3:45  | Breakout Session: AHA Research Agenda Feedback                         |
| 3:45 – 4:10  | Breakout Session Reports                                               |
| 4:10 – 4:40  | Closing Thoughts and AHA’s Research Intentions                         
                Mark Horowitz                                                      |