A Dog by Any Other Name Would Run Just as Slow
Computational and Hardware Complexity in Software Defined Radio

John S. Brunhaver II
School of Electrical Computer Energy Engineering
Fulton Schools of Engineering
Arizona State University

Feb 2nd 2020
Algorithms change to suit their hardware context

Who am I, to say this?
I am a digital hardware engineer

Tukwila (65nm)

Larrabee (45nm)

Beckton (32nm)

Kepler (28nm)

ASU-SNL-TC1 (12nm FinFet)
I started research with hw/sw co-design

ack. Kayvon Fatahalian

I Moved to DSL to RTL for Image Processing

- https://halide-lang.org/

---

John (ASU) Dogs, FPGAs, and SDRs! Oh My! Feb 2nd 2020 5 / 28
We demoed on FPGAs

- https://github.com/jingpu/Halide-HLS
- Xilinx Zynq-7000 SoC ZC702 Evaluation Kit
Currently we work on the general problem

- Given a set of code written by domain experts, can you
  - **Identify** kernels in an application
  - Procedurally **label** kernels, identifying equivalence
  - Discover the **taxonomy** of kernels
  - **Predict and arbitrage** accelerators and their organization
Currently we work on the general problem

- Given a set of code written by domain experts, can you
  - **Identify** kernels in an applications
  - Procedurally **label** kernels, identifying equivalence
  - Discover the **taxonomy** of kernels
  - **Predict and arbitrage** accelerators and their organization
  - Produce FPGA code
We currently focus on discovering kernels in code

Static

▶ CodeOcean – https://codeocean.com/capsule/2cb73b4e-11f9-4547-8fe3-4b4956d3d251/tree
▶ Git – https://github.com/ruhrie/TraceAtlas
We currently focus on discovering kernels in code

Static

- CodeOcean – https://codeocean.com/capsule/2cb73b4e-11f9-4547-8fe3-4b4956d3d251/tree
- Git – https://github.com/ruhrie/TraceAtlas
We currently focus on discovering kernels in code

Static

Dynamic

- CodeOcean – https://codeocean.com/capsule/2cb73b4e-11f9-4547-8fe3-4b4956d3d251/tree
- Git – https://github.com/ruhrie/TraceAtlas
We currently focus on discovering kernels in code

Static

Dynamic

Cluster

Data Flow

- CodeOcean – https://codeocean.com/capsule/2cb73b4e-11f9-4547-8fe3-4b4956d3d251/tree
- Git – https://github.com/ruhrie/TraceAtlas
FPGAs are just like puppies
FPGAs are just like puppies

Energy per Operation ($pJ$)

Area per Throughput ($mm^2/(op/ps)$)

- ASIC
- DSP Proc
- FPGA
- GPU
- Proc
- Systolic
- Vector

John (ASU) Dogs, FPGAs, and SDRs! Oh My! Feb 2nd 2020 9 / 28
FPGAs are just like puppies
Now you have a puppy – Systems, atoms, and accelerators

- System Pitfalls
  - Feed and Care of a Dog
- Structural FPGA Design
  - Atoms of traick
- Modelling Accelerator Structure
  - Putting the trick together
Now you have a puppy – Systems, atoms, and accelerators

- System Pitfalls
  - Feed -and- Care of a Dog
- Structural FPGA Design
  - Atoms of track
- Modelling Accelerator Structure
  - Putting the trick together
Understand your application’s composition
Understand your application’s composition
Optimize what matters
Optimize what matters

John (ASU)  Dogs, FPGAs, and SDRs! Oh My!  Feb 2nd 2020 12 / 28
Understand your sources of latency

Zynq UltraScale+ Device Technical Reference Manual UG1085 (v2.1)
August 21, 2019
Understand your sources of latency

Zynq UltraScale+ Device Technical Reference Manual UG1085 (v2.1)
August 21, 2019
Understand your sources of latency
Understand your sources of latency

For Dramatic effect only ...

Zynq UltraScale+ Device Technical Reference Manual UG1085 (v2.1)
August 21, 2019
Understand your sources of latency

For Dramatic effect only ...

▶ User - Blocking OS Call to use driver
Understand your sources of latency

For Dramatic effect only ...

- User - Blocking OS Call to use driver
- OS - Forced cache flush
Understand your sources of latency

For Dramatic effect only ...

- User - Blocking OS Call to use driver
- OS - Forced cache flush
- Driver - Forced data copy
Understand your sources of latency

For Dramatic effect only ...

- User - Blocking OS Call to use driver
- OS - Forced cache flush
- Driver - Forced data copy
- Driver - Send command to DMA
Understand your sources of latency

For Dramatic effect only ...

- User - Blocking OS Call to use driver
- OS - Forced cache flush
- Driver - Forced data copy
- Driver - Send command to DMA
- Driver - Poll DMA for complete
Understand your sources of latency

For Dramatic effect only ...

- User - Blocking OS Call to use driver
- OS - Forced cache flush
- Driver - Forced data copy
- Driver - Send command to DMA
- Driver - Poll DMA for complete
- Driver - Command Accel
Understand your sources of latency

For Dramatic effect only ...

- User - Blocking OS Call to use driver
- OS - Forced cache flush
- Driver - Forced data copy
- Driver - Send command to DMA
- Driver - Poll DMA for complete
- Driver - Command Accel
- Driver - Poll Accel for complete
Understand your sources of latency

For Dramatic effect only ...

- User - Blocking OS Call to use driver
- OS - Forced cache flush
- Driver - Forced data copy
- Driver - Send command to DMA
- Driver - Poll DMA for complete
- Driver - Command Accel
- Driver - Poll Accel for complete
- Driver - Command DMA
Understand your sources of latency

For Dramatic effect only ...

- User - Blocking OS Call to use driver
- OS - Forced cache flush
- Driver - Forced data copy
- Driver - Send command to DMA
- Driver - Poll DMA for complete
- Driver - Command Accel
- Driver - Poll Accel for complete
- Driver - Command DMA
- Driver - Poll DMA

Zynq UltraScale+ Device Technical Reference Manual UG1085 (v2.1)
August 21, 2019
Understand your sources of latency

For Dramatic effect only ...

- User - Blocking OS Call to use driver
- OS - Forced cache flush
- Driver - Forced data copy
- Driver - Send command to DMA
- Driver - Poll DMA for complete
- Driver - Command Accel
- Driver - Poll Accel for complete
- Driver - Command DMA
- Driver - Poll DMA
- OS - Restore
Understand your sources of latency

For Dramatic effect only ...

- User - Blocking OS Call to use driver
- OS - Forced cache flush
- Driver - Forced data copy
- Driver - Send command to DMA
- Driver - Poll DMA for complete
- Driver - Command Accel
- Driver - Poll Accel for complete
- Driver - Command DMA
- Driver - Poll DMA
- OS - Restore
Hide your latency

- Overlap transfer and operation
- Sequence operation in PL
- Execute and overlap many operations
Hide your latency

- Overlap transfer and operation
- Sequence operation in PL
- Execute and overlap many operations
- Avoid blocking, stream instead
  - Skip False memory barriers
  - True memory barriers
Hide your latency

- Overlap transfer and operation
- Sequence operation in PL
- Execute and overlap many operations
- Avoid blocking, stream instead
  - Skip False memory barriers
  - True memory barriers
### Understand your applications data-rate

<table>
<thead>
<tr>
<th>Kernel</th>
<th>Expression</th>
<th>Compute</th>
<th>Bandwidth</th>
<th>Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>Axpy</td>
<td>$y_i = \alpha x_i + b$</td>
<td>$N$</td>
<td>$N$</td>
<td>1</td>
</tr>
<tr>
<td>Inner Product</td>
<td>$z = \sum_{i=1}^{N} x_i y_i$</td>
<td>$N$</td>
<td>$N$</td>
<td>1</td>
</tr>
<tr>
<td>FIR-1D</td>
<td>$y_i = \sum_{l=1}^{M} \alpha_l x_{i-k}$</td>
<td>$NM$</td>
<td>$N$</td>
<td>$M$</td>
</tr>
<tr>
<td>FIR-2D</td>
<td>...</td>
<td>$N^2M^2$</td>
<td>$N^2$</td>
<td>$M^2$</td>
</tr>
<tr>
<td>FIR-3D</td>
<td>...</td>
<td>$N^3M^3$</td>
<td>$N^3$</td>
<td>$M^3$</td>
</tr>
<tr>
<td>Bubble Sort</td>
<td>$y_i : y_{i+1} \geq y_i \geq y_{i-1}$</td>
<td>$N^2$</td>
<td>$N$</td>
<td>$N$</td>
</tr>
<tr>
<td>GEMM</td>
<td>$z_{i,j} = \sum_{k=1}^{N} x_{i,k} y_{i,k}$</td>
<td>$N^3$</td>
<td>$N^2$</td>
<td>$N$</td>
</tr>
<tr>
<td>FFT</td>
<td>$X_k = \sum_{n=0}^{N-1} x_n \cdot e^{-\frac{j2\pi}{N} kn}$</td>
<td>$N\log N$</td>
<td>$N$</td>
<td>$\log N$</td>
</tr>
</tbody>
</table>
Understand your applications data-rate

<table>
<thead>
<tr>
<th>Resource</th>
<th>Count</th>
<th>Unit rate</th>
<th>Total bandwidth</th>
</tr>
</thead>
<tbody>
<tr>
<td>DRAM Ports</td>
<td>2</td>
<td>$128b \cdot 300MHz$</td>
<td>77 Gbps</td>
</tr>
<tr>
<td>BRAM Ports</td>
<td>$2k$</td>
<td>$32b \cdot 600MHz$</td>
<td>38 Tbps</td>
</tr>
<tr>
<td>DSP</td>
<td>$2k$</td>
<td>$3 \cdot 32b \cdot 600MHz$</td>
<td>114 Tbps</td>
</tr>
<tr>
<td>Reg</td>
<td>$0.5M$</td>
<td>$1 \cdot 600MHz$</td>
<td>300 Tbps</td>
</tr>
</tbody>
</table>

Roughly – Zynq UltraScale+ XCZU9EG-2FFVB1156 MPSoC
Exploit locality

- Avoid the memory wall
  - Registers mitigate band-limit in BRAM
  - BRAM mitigates band-limit in DRAM
  - Avoid writing back intermediates
Exploit locality

StencilA

Intermediate

StencilB
Exploit locality
Exploit locality

StencilA

Intermediate

StencilB

John (ASU) Dogs, FPGAs, and SDRs! Oh My! Feb 2nd 2020 16 / 28
Exploit locality

StencilA

Intermediate

StencilB
Exploit locality

StencilA

Intermediate

StencilB
Exploit locality

StencilA Intermediate StencilB
Exploit locality

StencilA  Intermediate  StencilB
Train actions then tricks – Understand FPGA structures

- System Pitfalls
  - Feed -and- Care of a Dog
- Structural FPGA Design
  - Atoms of trick
- Modelling Accelerator Structure
  - Putting the trick together
Train actions then tricks – Understand FPGA structures

- System Pitfalls
  - Feed -and- Care of a Dog
- Structural FPGA Design
  - Atoms of trick
- Modelling Accelerator Structure
  - Putting the trick together
Think structurally

Resources
- BRAM - 2× 18kb / 36b RW
- DSP - 2× 24b MAC
- LUTs - 300× 5:2 fx
- Regs - 300× 1b state
- Routing
Think structurally

Resources
- BRAM - 2 × 18kb / 36b RW
- DSP - 2 × 24b MAC
- LUTs - 300 × 5:2 fx
- Regs - 300 × 1b state
- Routing

Minor misconceptions

- Not a Verilog accelerator
- Slices, not LUTs and Regs
- Wires are expensive, not gates
- Global wires are worse

John (ASU) Dogs, FPGAs, and SDRs! Oh My! Feb 2nd 2020
Think structurally

Resources
- BRAM - 2× 18kb / 36b RW
- DSP - 2× 24b MAC
- LUTs - 300× 5:2 fx
- Regs - 300× 1b state
- Routing

Minor misconceptions
- Not a Verilog accelerator
Think structurally

Resources
- BRAM - 2× 18kb / 36b RW
- DSP - 2× 24b MAC
- LUTs - 300× 5:2 fx
- Regs - 300× 1b state
- Routing

Minor misconceptions
- Not a Verilog accelerator
- Slices, not LUTs and Regs
Think structurally

- **Resources**
  - BRAM - 2× 18kb / 36b RW
  - DSP - 2× 24b MAC
  - LUTs - 300× 5:2 fx
  - Regs - 300× 1b state
  - Routing

- **Minor misconceptions**
  - Not a Verilog accelerator
  - Slices, not LUTs and Regs
  - Wires are expensive, not gates
Think structurally

Resources
- BRAM - 2× 18kb / 36b RW
- DSP - 2× 24b MAC
- LUTs - 300× 5:2 fx
- Regs - 300× 1b state
- Routing

Minor misconceptions
- Not a Verilog accelerator
- Slices, not LUTs and Regs
- Wires are expensive, not gates
- Global wires are worse
Pipeline your DSP

UltraScale Architecture DSP Slice User Guide UG579 (v1.9) September 20, 2019
Pipeline your DSP

UltraScale Architecture DSP Slice User Guide UG579 (v1.9) September 20, 2019

Figure 2-1: Detailed DSP48E2 Functionality

*These signals are dedicated routing paths internal to the DSP48E2 column. They are not accessible via general-purpose routing resources.
Pipeline your BRAM
Maximize resource utilization

- **Assets:**
  - DSP - Rate, Location
  - BRAM - Capacity, Ports, Bandwidth, Location
  - DRAM - Bandwidth, Ports

- **Goals:**
  - Spatial utilization (70 – 90%)
  - Temporal utilization (> 99%)
  - Minimize regret
Plan a trick from atoms – Predict Accelerator Structure

- System Pitfalls
  - Feed -and- Care of a Dog
- Structural FPGA Design
  - Atoms of trick
- Modelling Accelerator Structure
  - Putting the trick together
Plan a trick from atoms – Predict Accelerator Structure

- System Pitfalls
  - Feed and Care of a Dog
- Structural FPGA Design
  - Atoms of trick
- Modelling Accelerator Structure
  - Putting the trick together
In Place FFT

- BRAM 16Kb
- BRAM 16Kb
- BRAM 16Kb
- BRAM 16Kb

xBar 4 to 4 x2

Radix-2
Butter
Fly

128b
32b
32b
32b
Make a "good" FOSS Verilog library

- Utilize best practices from software
  - Compact API
  - Parametric calls
  - Well documented
  - Unit tested
  - Benchmarked
Putting it all together

- Optimize what matters
- Hide your latency
- Exploit locality
- Think structurally
Putting it all together

- Optimize what matters
- Hide your latency
- Exploit locality
- Think structurally
- Algorithms change to suit their hardware context
Questions, Thoughts, Concerns, Hopes, Dreams, Desires?