ORCA
FPGA-Optimized

© 2016 VectorBlox Computing Inc.
ORCA FPGA-Optimized

Tiny, Low-Power FPGA
3,500 LUT4s
4 MUL16s
< $5.00
~ 20MHz

ISA: RV32IM
hw multiply, sw divider
< 2,000 LUTs
What is ORCA?

• Family of RISC-V implementations
  – Highly parameterized
  – Ideally suited for FPGAs
  – Portable across FPGA vendors
  – BSD license open source hardware
Why ORCA?

• Many reasons
  – Orcas travel in pods: family of many sizes
  – Orcas are native to Vancouver

• ORCA – many possible backronyms
  – ORCA RISC-V Computer Architecture
  – ORCA Reconfigurable CPU Architecture
  – Optimized RISC-V CPU Architecture
ORCA: Multiple FPGA Vendors

- **Altera**
  - Drop-in Qsys replacement for Nios II/f
  - Avalon I / D masters

- **Lattice**
  - Wishbone I / D masters

- **Xilinx, Microsemi**
  - Coming soon
ORCA RISC-V RV32I on Different FPGAs

<table>
<thead>
<tr>
<th></th>
<th>LATTICE ICE40 Ultra™</th>
<th>ALTERA Cyclone® IV</th>
<th>ALTERA Stratix® V</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area</td>
<td>2008 LUT4</td>
<td>1623 LUT4</td>
<td>541 ALMs</td>
</tr>
<tr>
<td>Fmax</td>
<td>22 MHz</td>
<td>109 MHz</td>
<td>244 MHz</td>
</tr>
<tr>
<td>DMIPS</td>
<td>n/a</td>
<td>79 MIPS</td>
<td>212 MIPS</td>
</tr>
<tr>
<td>DMIPS/MHz</td>
<td>n/a</td>
<td>0.73</td>
<td>0.87</td>
</tr>
</tbody>
</table>
## ORCA vs Other RISC-V

<table>
<thead>
<tr>
<th></th>
<th>ORCA RV32IM</th>
<th>Z-scale RV32IM</th>
<th>PicoRV RV32I</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Area</strong></td>
<td>2353 LUT4 (Cyclone IV, 60nm)</td>
<td>2678 LUT4 (Spartan 6, 45nm)</td>
<td>2949 LUT4 (Cyclone IV, 60nm)</td>
</tr>
<tr>
<td><strong>Fmax</strong></td>
<td>125 MHz</td>
<td>33 MHz</td>
<td>127 MHz</td>
</tr>
<tr>
<td><strong>DMIPS</strong></td>
<td>122 MIPS</td>
<td>44 MIPS</td>
<td>39 MIPS</td>
</tr>
<tr>
<td><strong>DMIPS/MHz</strong></td>
<td>0.98 (measured)</td>
<td>1.35 (claimed)</td>
<td>0.31 (claimed)</td>
</tr>
</tbody>
</table>
# ORCA RISC-V vs FPGA CPUs

<table>
<thead>
<tr>
<th></th>
<th>ORCA RV32IM</th>
<th>Altera Nios II/f</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area</td>
<td>2353 LUT4 (Cyclone IV)</td>
<td>2678 LUT4 (Cyclone IV)</td>
</tr>
<tr>
<td>Fmax</td>
<td>125 MHz</td>
<td>140 MHz</td>
</tr>
<tr>
<td>DMIPS</td>
<td>122 MIPS</td>
<td>163 MIPS</td>
</tr>
<tr>
<td>DMIPS/MHz</td>
<td>0.98 (measured)</td>
<td>1.16 (claimed)</td>
</tr>
</tbody>
</table>

ORCA RISC-V vs FPGA CPUs
RISC-V: Architecture Space

- **Width (3 choices)**
  - 32, 64, 128 bits

- **Instruction Set (9 binary options, 2^9 choices)**
  - Minimum: I
  - Binary options: M, A, F, D (== G), Q, L, B, T, P

- **Instruction Encoding (2 choices)**
  - C

- **Architecture Space 3 x 2^10 = 3072 possibilities**
ORCA Implementation Space

- Logic design (12 choices)
  - Multiplier (sw, hw)
  - Divider (sw, hw)
  - Shifter (1-cycle, 8-cycles, 32-cycles)
- Counters (3 choices)
  - 0, 32, or 64 bits
- Pipelining (2 choices)
  - 4 or 5 stages
- Forwarding (2 choices)
  - ALU only
  - ALU + other units

Implementation space $12 \times 3 \times 2 \times 2 = 144$ possibilities

Overall arch. x impl. = $3072 \times 144 / 2 = 221,184$ possibilities
Huge Design Space

- **Implementation on ASIC**
  - Need to choose one design point in the architecture + implementation space
  - Benefit: user has no choice
  - Problem: compromise across many applications

- **Implementation on FPGA**
  - Can have fully parameterized design
  - User can choose best architecture + implementation according to application
  - Benefit: good performance, area
  - Problem: overwhelming design space
32b vs 64b Counters

Execution Time:
- Faster
- Slower

LUT6 count:
- Smaller
- Bigger

32b counters
- no counters
- 64b counters
4 vs 5 Pipeline Stages

Execution Time vs LUT4 count

- 4 pipeline stages
- 5 pipeline stages
FPGA \rightarrow ASIC

but

ASIC !\rightarrow FPGA

good FPGA implementation
\rightarrow often leads to good ASIC implementation

good ASIC implementation
\rightarrow often leads to poor FPGA implementation
Register File

• Discrete FFs: inefficient

- 32 cpu registers x 32 b = 1024 FFs
- 32 mux32 = 32 x 11 LUT4 = 352 LUT4s

• Note: muxes are costly, must avoid!!!
Register File Implications

- Block RAMs: dual ported, registered output
  - Use 1 RD, 1 WR port
  - Use data-out FFs as pipeline FFs
  - Needs external data-forwarding

RAM cannot forward WB data to OPERAND data internally
ORCA Datapath

Fetch

Instr. Fetch

Instr. Avalon / Wishbone

Decode

RF1

RF2

Forward (can be collapsed into Ex/Mem stage)

Ex/Mem

ALU / BR / SLT

CSR

LD/ST

Data Avalon / Wishbone

WB
Some FPGA Suggestions

• RV32E spec
  – Reduced # registers saves nothing in FPGAs
  – Divide is expensive

• Software
  – Beware, shifts may be 1b/cycle (slow)

• Privileged Arch spec
  – Too many CSRs, 64b counters too big
    • Increases pressure on multiplexers
  – Suggest small / med / full versions
    • No “official” rules on what to include/exclude to reduce size
    • Eg, hypervisors not likely to run on FPGAs
Conclusions

• ORCA RISC-V family is free, portable, FPGA-optimized
  – FPGA and ASIC optimizations are different
    • FPGA architecture dictates certain design choices
  – Some RISC-V decisions are “unconsciously” aimed towards ASIC implementation
    • These do not lead to good FPGA implementations
    • But good FPGA choices lead to good ASICs
Free FPGA Hardware!

- **Today only:** Lattice donating FPGA boards for RISC-V users

  - **ORCA RV32I system** ~2000 LUTs
    - [http://www.github.com/VectorBlox/risc-v](http://www.github.com/VectorBlox/risc-v)
    - About 1500 LUTs available for user I/O (e.g., UART)
LUNCHTIME

So Long, and Thanks for All the Fish!!