NVIDIA RISC-V Story

4th RISC-V Workshop 7/2016
Introduce NVIDIA falcon CPU
Why a new CPU?
Introduce NV-RISCV
Falcon = FAst Logic CONtroller

Introduced over 10 years ago, and used in >15 different hardware engines today

Design for flexibility
Design for long memory latency
Design for low area
Design for security
Why Falcon Next Gen?

- New use cases requiring more horsepower & feature
  - Wide addressing range
  - More performance
  - Not limit to code size
  - Rich OS support

- Falcon has limits
  - Small addressing range
  - Poor performance (0.67DMIPS/Mhz, 1.4Coremark/Mhz)
  - No D$
  - No rich OS support
Falcon Next Gen - Options

**Buy**
- ARM (A,R family)
- Synopsys (ARC family)
- MIPS
- Cadence

**Build**
- Improve falcon
- Move to a new ISA (And this is when RISC-V came into the picture..)
**CPU comparison**

**Conclusion** -
- RISC-V is the right direction to next generation of ISA
- Build our own implementation of RISC-V core

<table>
<thead>
<tr>
<th>Item</th>
<th>Requirement</th>
<th>ARM A53</th>
<th>ARM A9</th>
<th>ARM R5</th>
<th>SNPS HS</th>
<th>RISC-V Rocket</th>
<th>Falcon (improved)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core perf</td>
<td>&gt;2x falcon</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Area (16ff)</td>
<td>&lt;0.1mm^2</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Security</td>
<td>Yes</td>
<td>TZ</td>
<td>TZ</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>TCM</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>L1 I/D $</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Addressing</td>
<td>64bit</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Extensible ISA</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Safety (ECC/Parity)</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Functional Simulation model</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Introducing NV-RISCV
## Summary: Falcon -> NV RISC-V

<table>
<thead>
<tr>
<th>Feature</th>
<th>Falcon</th>
<th>NV-RISCv</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISA</td>
<td>Falcon-ISA</td>
<td>RISCV-RV64</td>
</tr>
<tr>
<td>Address Width</td>
<td>32/24</td>
<td>64</td>
</tr>
<tr>
<td>Data width</td>
<td>32</td>
<td>64</td>
</tr>
<tr>
<td>GPR Num.</td>
<td>16</td>
<td>32</td>
</tr>
<tr>
<td>Stage</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>Micro-arch</td>
<td>In-order issue</td>
<td>In-order issue</td>
</tr>
<tr>
<td></td>
<td>out-of-order exec</td>
<td>out-of-order exec</td>
</tr>
<tr>
<td></td>
<td>out-of-order WB (diff regs)</td>
<td>in-order WB</td>
</tr>
<tr>
<td>WAW Hazard</td>
<td>Stall</td>
<td>ROB</td>
</tr>
<tr>
<td>Cache</td>
<td>No D cache</td>
<td>I/D configurable</td>
</tr>
<tr>
<td>TCM</td>
<td>I/D configurable</td>
<td>I/D configurable</td>
</tr>
<tr>
<td>Prediction</td>
<td>static</td>
<td>BTB/BHT/RAS</td>
</tr>
<tr>
<td>Load-store</td>
<td>In-order</td>
<td>Load out-of-order</td>
</tr>
<tr>
<td>Memory protection</td>
<td>No</td>
<td>MPU</td>
</tr>
<tr>
<td>Address mapping</td>
<td>TCM tagging</td>
<td>Base and bound</td>
</tr>
</tbody>
</table>
Falcon Next Gen with RISCV

- RISCV plugged-in as 2\textsuperscript{nd} core
  - Back compatibility on interface, easy to integrate
  - Isolation between security and non-security applications
### NV-RISCV Core perf/area

#### Area data under 16ff

<table>
<thead>
<tr>
<th>Core (TSMC)</th>
<th>Falcon (Today)</th>
<th>RISCV rocket chip</th>
<th>NV-RISCV</th>
<th>BOOM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dhrystone (no inline)</td>
<td>0.67</td>
<td>1.72</td>
<td>1.9~2.0</td>
<td>N/A</td>
</tr>
<tr>
<td>EEMBC Core Mark</td>
<td>1.4</td>
<td>2.3</td>
<td>~2.5</td>
<td>3.91</td>
</tr>
<tr>
<td>Core Area(mm²)</td>
<td>0.03</td>
<td>0.055 (no FPU, no L2)</td>
<td>0.05~0.06</td>
<td>N/A</td>
</tr>
<tr>
<td>Frequency(GHz)</td>
<td>1.5</td>
<td>~1</td>
<td>&gt;1.5</td>
<td>N/A</td>
</tr>
</tbody>
</table>
Cache design to tolerate large latency

- Configurable cache size/line size/associativity/write policy
- Cache Optimizations
  - Store Buffer
  - Write merging
  - Line-fill Buffer
  - Victim Buffer
  - Stream buffer
  - SW pre-fetch (future)
  - L2 (future)
  - Banked cache (future)
- I/DTCM
D$ perf - btree (8k nodes)

Total cycles (8k nodes, 64KB D Cache), less is better

- RISCV: 371.35%
- Falcon: 220.53%
- RISV optimized: 169.02%
- Falcon ideal: 100.00%

Total cycles (normalized)
RISC-V - Area of Interests to NV

- **Tool chain**
  - Tool for automotive - To meet ISO26262/ASIL-D or SIL3 requirements
  - Tool for debug - To debug ucode on silicon/FPGA/emulator
  - Tool for performance tuning - For ucode profiling and tuning
  - Tool for flexibility - That users can easily customize ISA
  - Other compiler features - ILP32/LP64

- **Security**
  - Crypto instruction & extensions

- **More instructions**
  - Cache instruction - pre-fetch/invalidate/flush...
Conclusion

- Falcon is NVIDIA proprietary control processor
- New use-cases require more feature and performance from falcon
- It is hard to improve the current CPU/ISA to meet all new requirements
- We evaluated different options in the market, result showed that RISC-V is overall best choice as next generation of falcon
- We will build a new core from RISC-V ISA
Thank You!