How much score could a CoreScore score if a CoreScore could score cores?

Spring 2022 RISC-V week - Paris 2022.05.04
Who are we?

Klas Nordmark
FPGA & Embedded SW Consultant at 2550 Engineering

Olof Kindgren
FPGA & Embedded SW Consultant at Qamcom Research & Technology
Features

- RV32I
- Wishbone data bus
- Wishbone instruction bus
- CFU extension interface
- Formally verified with riscv-formal
- BSD licensed
- + optionally
  - CSR extension
  - A bit of the privilege spec (enough to run Zephyr)
  - M extension
  - C extension
Zeeshan Rafique
M extension, GSoC 2021

Abdul Wadood
C extension LFX 2022
What's a bit-serial CPU?
Bit-serial

Write 0xA0 | 0x33 = 0xB3
Bit-serial

Skriv $0xA0 \mid 0x33 = 0xB3$

1010000 0
0011001 1

$\downarrow$

1
Bit-serial

Skriv 0xA0 | 0x33 = 0xB3

\[ \begin{array}{c|c}
101000 & 0 \\
001100 & 1 \\
\end{array} \]
Bit-serial

Skriv $0xA0 | 0x33 = 0xB3$

$$
\begin{array}{c}
10100 0 \\
00110 0
\end{array}
$$
Bit-serial

Skriv 0xA0 | 0x33 = 0xB3
Bit-serial

Skriv $0xA0 | 0x33 = 0xB3$

\[
\begin{array}{c}
  101 \\
  001 \\
\end{array}
\]

\[
\begin{array}{c}
  1 \\
  0011 \\
\end{array}
\]
Bit-serial

Skriv $0xA2 + 0x33 = 0xD5$
Bit-serial

Skriv $0xA0 \mid 0x33 = 0xB3$

```
0101001 0
0011001 1
```

0 + 1 = 1

Diagram:
- Square block with inputs: 0101001, 0011001
- Output: 1
Bit-serial

Skriv \(0xA0 \ | \ 0x33 = 0xB3\)
Bit-serial

Kriv 0xA0 | 0x33 = 0xB3
Bit-serial
<table>
<thead>
<tr>
<th></th>
<th>Minimal</th>
<th>Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>LUT</td>
<td>FF</td>
</tr>
<tr>
<td>Lattice iCE40</td>
<td>198</td>
<td>164</td>
</tr>
<tr>
<td>AMD Artix-7</td>
<td>125</td>
<td>164</td>
</tr>
<tr>
<td>Intel Cyclone 10</td>
<td>239</td>
<td>164</td>
</tr>
</tbody>
</table>
Documentation
serv_alu handles alu operations. The first input operand (A) comes from i_rs1 and the second operand (B) comes from i_rs2 or i_imm depending on the type of operation. The data passes through the add/sub or bool logic unit and finally ends up in o_rd to be written to the destination register. The output o_cmp is used for conditional branches to decide whether or not to take the branch.

The add/sub unit can do additions A+B or subtractions A-B by converting it to A+B+1. Subtraction mode (i_sub = 1) is also used for the comparisons in the shift and conditional branch instructions. The +1 used in subtraction mode is done by preloading the carry input with 1. Less-than comparisons are handled by converting the expression A<B to A-B<0 and checking the MSB, which will be set when the result is less than 0. This however requires sign-extending the operands to 33-bit inputs. For signed operands (when i_cmp_sig is set), the extra bit is the same as the MSB. For unsigned, the extra bit is always 0. Because the ALU is only active for 32 cycles, the 33rd bit must be calculated in parallel to the ordinary addition. The result from this operation is available in result_lt. For equality checks, result_eq checks that all bits are 0 from the subtraction.
• Minimal reference platform
• Available for 30+ FPGA boards
• Runs Zephyr OS
• Minimal reference platform
• Available for 30+ FPGA boards
• Runs Zephyr OS

Servant
Serving

- SoClet
- RF in RAM
Use cases

• Replace complex FSM (e.g. DDR init, logging, status...)
• Sensor data aggregation
• Replace 8-bit MCUs
• Replace threaded apps
• Benchmarking
an award-giving benchmark for FPGAs and their synthesis/P&R tools

https://github.com/oloef/corescore
an award-giving benchmark for FPGAs
and their synthesis and P&R tools

<table>
<thead>
<tr>
<th>Board</th>
<th>CoreScore</th>
</tr>
</thead>
<tbody>
<tr>
<td>vcu128</td>
<td>6000</td>
</tr>
<tr>
<td>intel_s10gx_devkit</td>
<td>5600</td>
</tr>
<tr>
<td>vcu118</td>
<td>5087</td>
</tr>
<tr>
<td>haps_dx7</td>
<td>3040</td>
</tr>
<tr>
<td>intel_a10gx_devkit</td>
<td>2900</td>
</tr>
<tr>
<td>de5_net</td>
<td>1568</td>
</tr>
<tr>
<td>sturdypeak</td>
<td>1152</td>
</tr>
<tr>
<td>hpc_k7</td>
<td>1024</td>
</tr>
<tr>
<td>hpc_ku</td>
<td>1024</td>
</tr>
<tr>
<td>genesys2</td>
<td>967</td>
</tr>
<tr>
<td>kc705</td>
<td>960</td>
</tr>
<tr>
<td>zcu106</td>
<td>940</td>
</tr>
<tr>
<td>polarfireval</td>
<td>882</td>
</tr>
</tbody>
</table>
OpenMPW is a Google-financed initiative by Efabless for doing free prototype runs of open source ASICs in the Skywater 130nm process.

Submissions are built using the FOSS Openlane flow and are required to be surrounded by the Caravel harness, including a picorv32 with a Wishbone interface that the user design can connect to.
Subservient is a variant of Serving exposing an SRAM interface and a Wishbone debug interface that can read or write the SRAM when Subservient is kept in debug mode with a separate pin.

We have sent in two slightly different variants, in order to meet timing with later versions of Openlane after the presumed failure of MPW-2.
Subservient – SERV in OpenMPW

User project area of 2920 by 3520 micrometers

Subservient synthesized together with 512 bytes of DFFRAM about 1000 by 1000 micrometers, 25 MHz clock

Implies a corescore of six or so. Not very good...
Subservient – SERV in OpenMPW

Removing the memory both significantly decreases gate count and eases up the place and route problem, allowing higher density.

Subservient without memory here about 160 by 160 micrometers

If memory was out of the picture completely, we'd have a Corescore of nearly 400!
Subservient – SERV in OpenMPW

Using OpenRAM has been problematic for us.

1kbyte, 1 byte wide ready-made macro for sky130A is about 460 micrometers wide.

With one such macro for each smaller Subservient, we'd look at about 20 cores in the user project area...

Smaller memories? Shared memories?
Subservient – SERV in OpenMPW

We have not yet received the first batch of chips and can't yet confirm if SERV has been successfully ASIC-proven.
Future work

DSRV, QERV
More extensions
FPGA with hard SERV for each SRAM
Recreate SERV in Logisim, Minecraft, 7400 chips, tubes...
Thank you for your time

Subservient
https://github.com/klasnordmark/subservient_wrapped

Klas Nordmark
linkedin.com/in/klas-nordmark-33a97244
@knordmark89
https://2550.engineering

Olof Kindgren
linkedin.com/in/olofkindgren
@OlofKindgren
https://qamcom.com