Compact CORDIC accelerator implementation for embedded RISC-V core

→ Codasip Studio and RISC-V Processors

Begin with a standard core
- Embedded and application cores
- High quality, production-ready
- Fully RISC-V compliant

Differentiate with Codasip Studio
- Configure / Customize
- Using CodAL architecture description language

→ CORDIC accelerator implemented with CodAL

```plaintext
semantics {
  angle = rfi_gpr_read(src);
  cos = CORDIC_GAIN;
  sin = 0;
  for (shift=0; shift < ITERATIONS; shift++) {
    if (angle<0) {
      cos = sin >>> shift;
      sin = cos >>> shift;
      angle = tan(shift);
    } else {
      cos = sin >>> shift;
      sin = cos >>> shift;
      angle = tan(shift);
    }
  }
  rfi_gpr_write(dst, (cos = sin));
  codasip_ino_clock_cycle(16);
};
```

→ How customization affects the PPA

1 custom instruction call to start “CORDIC” flow
16 cycles to get the result
16-bit fixed-point results representation

<table>
<thead>
<tr>
<th>(TSMC 28nm)</th>
<th>RISC-V(L31)</th>
<th>+ CORDIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area, a.u.</td>
<td>100%</td>
<td>104.4%</td>
</tr>
<tr>
<td>Performance gain</td>
<td>1x</td>
<td>24.3x</td>
</tr>
<tr>
<td>Energy consumption</td>
<td>100%</td>
<td>7.4%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Design time, lines of code in CodAL</th>
<th>Lines of code in Verilog</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 person-days</td>
<td>210</td>
</tr>
<tr>
<td></td>
<td>600 (~3x)</td>
</tr>
</tbody>
</table>