A gem5-based CVA6 Framework for Microarchitectural Pathfinding

Pierre Ravenel\textsuperscript{1,2} Arthur Perais\textsuperscript{2} Benoît Dupont de Dinechin\textsuperscript{1} Frédéric Pétrot\textsuperscript{2}

\textsuperscript{1}Kalray\textsuperscript{2} Univ. Grenoble Alpes, CNRS, Grenoble INP, TIMA

Overview

The big picture

- Designing well-balanced general purpose processor targeting ASIC implementation is arduous
- \(\rightarrow\) Software timing models enable fast design space exploration in pathfinding phase

The goal

- Create fast timing simulation environment to drive microarchitectural improvements for the CVA6
- \(\rightarrow\) Depict correlation flow ensuring timing model keeps projecting meaningful performance numbers

Model Accuracy

Idea: find a compromise between cycle accuracy and microarchitectural abstraction level

In Practice

Principle: 3 steps modelisation

(1) System, (2) Front end and (3) Back end

Validation and Results

Validation methodology

- We created a micro-benchmark suite with asymptotic behavior
- Read-after-Write dependencies example

```
# loop/loop_raw.S
_start:
addi t0 , zero , 1000  # Initialise loop
addi t1 , zero , 0

loop:               # Begin loop
    #
    addi a0 , a5 , 4  # write a0,
    # \ RAW
    addi a1 , a0 , 4  # write a1, read a0
    # \ RAW
    addi a2 , a1 , 4  # read a1
    addi a3 , a2 , 4  # ...
    addi a4 , a3 , 4
    addi a5 , a0 , 4
    #
    addi t1 , t1 , 1  # End loop
bge t0 , t1 , loop
```

Polybench execution times

- Very close to those of CVA6 RTL simulation
- Albeit obtained 1000 times faster

Perspectives

- Constructed CVA6 performance model highlights design bottlenecks while facilitating design exploration
- With a bottom-up approach, we identified performance losses in corner cases
- With a top-down approach, next step is to improve the backend with value-prediction

We designed a CVA6 cpu model in gem5