The RISC-V Week, Paris, October 1-3

An Out-of-Order RISC-V Core Developed with HLS

Bernard Goossens and David Parello

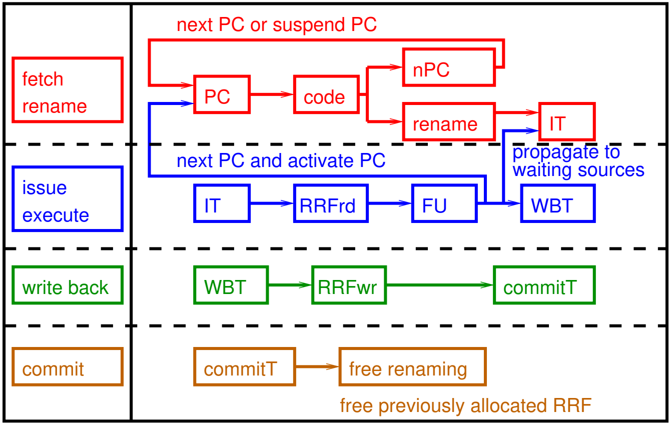
Université de Perpignan Via Domitia, DALI-LIRMM

Outline

Section 1

The Out-of-Order RISC-V Core Microarchitecture

The 4-stages pipeline



- Four concurrent stages, No cache, Flat memory, No branch predictor
- Fetch blocked by cond. or indir. branch decode, unblocked by its computation
- Compatible with Multicycles Functional Units
- RISCV32IM ISA

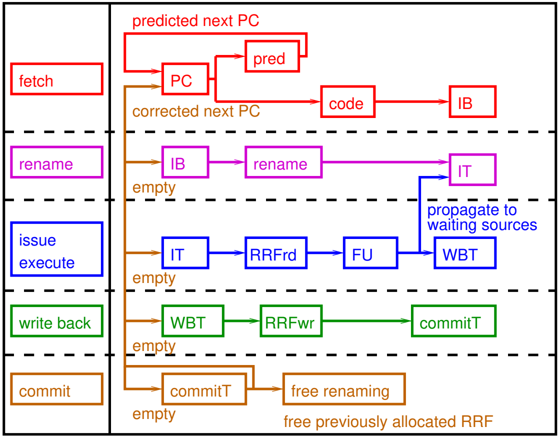
The HLS code

- Xilinx Vivado_hls code written in two weeks by one person
- 9 C files, 6 header files, less than 4000 lines
- Optimized with Vivado_hls constraints to fit each stage in the by default 10ns cycle
  - Multiple ports on BRAM variables (e.g. table of architectural register renaming)
  - Full unrolling of arrays and loops (e.g. IT table)
  - Elimination of dependencies (e.g. PC write in execute stage and read in fetch stage)
- Estimation of speed and area on Catapult (Simon Rokicki, Cairn team at Inria)
  - ASIC, technology 28nm FDSOI, 700 MHz : 50 300 μm2 (Comet in-order = 8168 μm2)
  - FPGA, Xilinx ZCU106, 100Mhz : 9146 LUT, 11215 FF, 970 Mux (Comet = 2032, 1503, 260)
  - 64x32-bit renaming registers, 64x99-bit entries instruction table, 64*15-bit ROB, 32*7-bit renaming table, 64*6-bit free rr list, 64*6-bit free it list : 10336 FF

Section 2

The RISC-V OOO core : an educational kernel tool to be expanded

Adding speculation



- A branch predictor
- Allows to separate decoding and renaming from fetch
- Can be enhanced with checkpointing (correct PC and reconstruct renaming at WB)

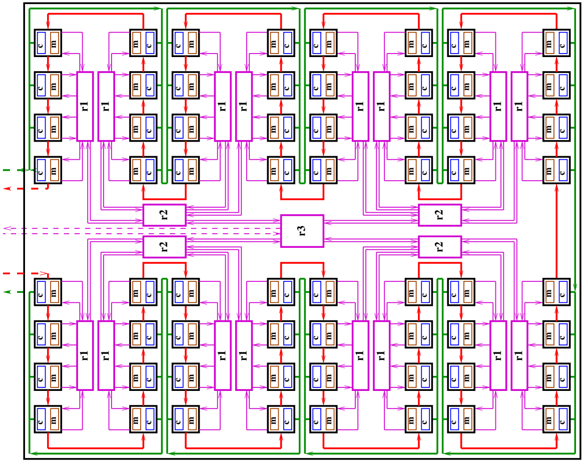
Other enhancements

- Increasing the superscalar degree
- Adding multicycle execution pipelines (e.g. pipelined multiplier)
- Hierarchizing memory
- Adding an FPU
- Multithreading

Section 3

The RISCV OOO core : the main building block of the LBP processor

The Little Big Processor (LBP) : a manycore



- An extension of the RISC-V OOO core is the building block of the 64-core LBP processor
- The LBP processor is a manycore multithreaded design with an inter-thread communication capability
- The LBP processor is able to catch arbitrary distant ILP

LBP captures distant ILP

- LBP divides a run into threads
- Threads are ordered to reflect the sequential semantic of the application
- Threads are run in parallel
- Threads are spreaded on the LBP harts (fork/join model)
- The main obstacles to inter-thread ILP capture are removed (no stack, no control flow dependency, new register file)
- The available ILP is the sum of the threads individual ILPs
- The ILP is proportional to the data size

LBP is a deterministic processor

- No interrupt
- A single application
- Perfect isolation (no interference from the external world or between concurrent applications)
- If the appication is isolated from the external world, it is cycle deterministic
- Otherwise, it is run in a deterministic partial order

64-core LBP in HLS

- Xilinx Vivado_hls code written in two months by one person
- 10 C files, 7 header files, less than 7500 lines
- Tested with the Vivado simulator (e.g. Parallelized tiled matrix multiplication)
- Implementation on ZCU106 in progress
- estimation : 3500 FF (0.75%) (64-core LBP = 48%)
- 16x32-bit renaming registers, 16x91-bit entries instruction table, 16*13-bit ROB, 32*5-bit renaming table, 16*4-bit free rr list, 16*4-bit free it list : 2464 FF

Section 4

Conclusion

Conclusion

- A RISC-V OOO processor implemented from HLS tools
- Code available on demand (tared source files, no installation required)
- HLS is now adult and can be used in replacement of VHDL/Verilog, at least for prototypes