Automatic Micro-Architecture Exploration and Synthesis for RISC-V CPUs

Jean-Michel Gorius, Steven Derrien, Simon Rokicki

Univ Rennes, Inria, CNRS, IRISA

{jean-michel.gorius, steven.derrien, simon.rokicki}@irisa.fr

Designing a RISC-V CPU *should* be as simple as writing an Instruction Set Simulator

Introduction

Context: Increasing need for customizable architectures for embedded applications.

Problem: Micro-architectural design is tedious and error-prone, how do we make such customizations available to everyone?

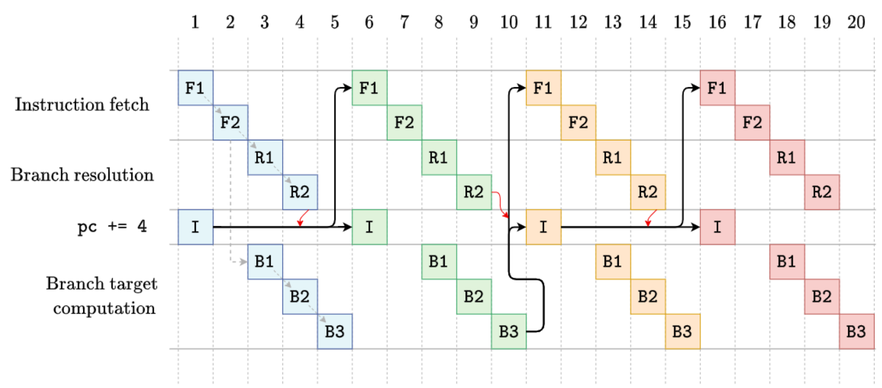
Our approach: Leverage High-Level Synthesis to **synthesize micro-architectures** from a single instruction set simulator in C.

Synthesizing In-Order Pipelined Instruction Set Processors

```
while(1) {
  unsigned int ir = fetch(mem, pc);
  pc += 4;

  struct decode_info dc = decode(ir);
  unsigned int rs1 = x[dc.rs1];
  unsigned int rs2 = x[dc.rs2];

  switch(opcode(dc)) {
  case RISCV_ADD:
    x[dc.rd] = add(rs1, rs2);
    break;
  case RISCV_MUL:
    x[dc.rd] = mul(rs1, rs2);
    break;
  case RISCV_JAL:
    pc = pc + dc.simm_J;
    break;
  case RISCV_LD:
    if(dc.funct3 == RISCV_LD_LB)
        x[dc.rd] = mem[rs1 + dc.imm_S];
    else if(/* ... */)
      // ...
    break;
  // ...
  }
}
```



What we have with traditional loop pipelining

Intra-iteration dependencies

Inter-iteration dependencies

Key takeaway

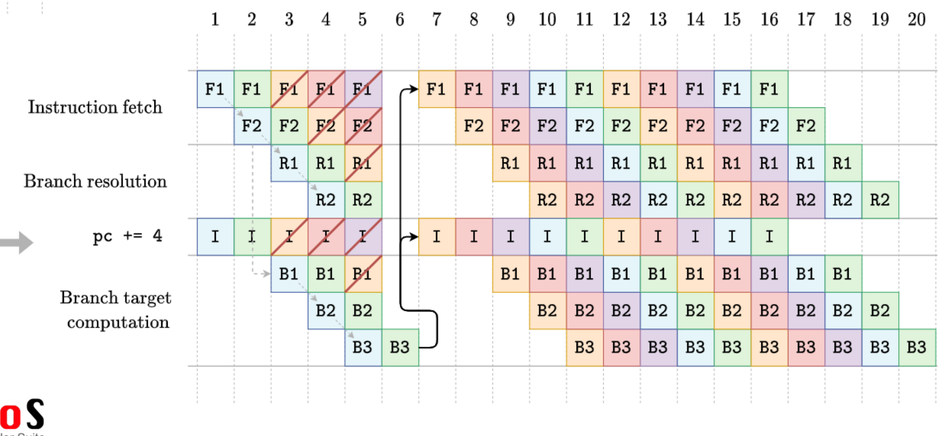
Designing a CPU should be as **simple** as writing an Instruction Set Simulator in C.

What we want

SpecHLS toolchain

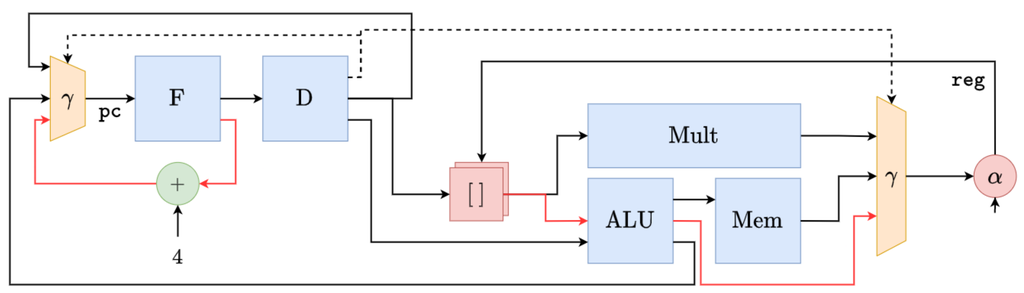
- Fully automated design toolchain
- Traditional HLS toolchain as a backend
- Based on Speculative Loop Pipelining [Derrien'2020]

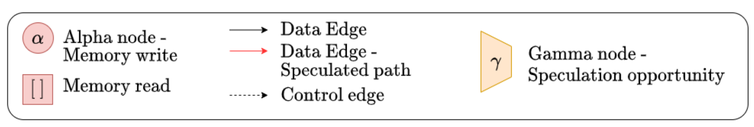
SpecHLS



GeCoS Generic Compiler Suite

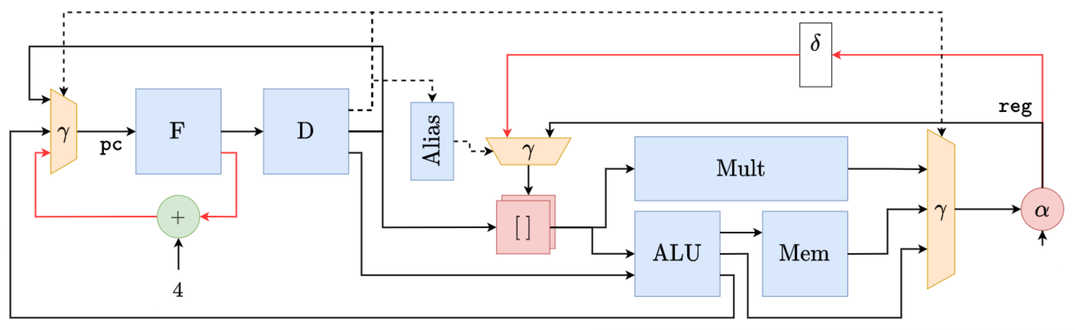
Speculation configuration exploration





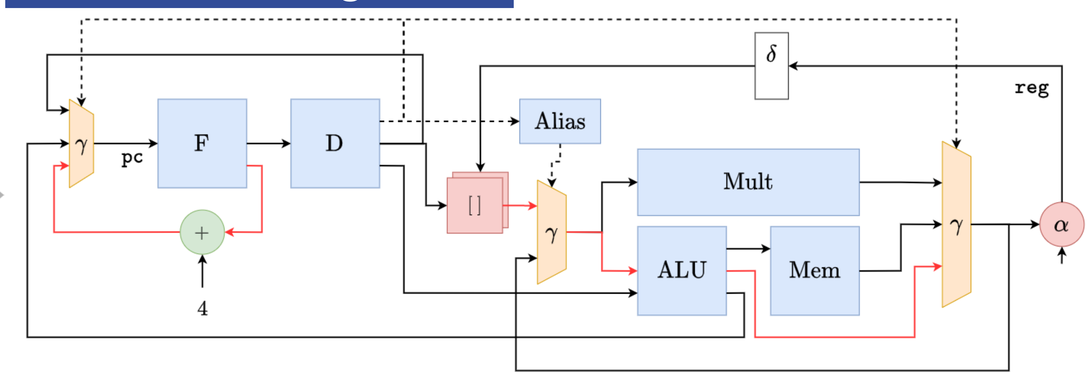
Exploring speculation opportunities

- Speculation opportunities derived from unbalanced paths in the input
- Design space exploration leveraging *intertwined speculations* to obtain the best designs



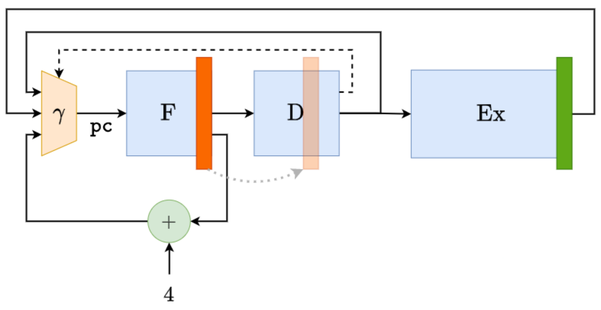
Pipeline stall

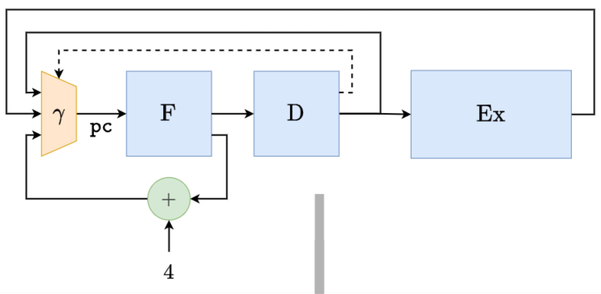
Forwarding



Pipeline depth exploration

3-stage pipeline

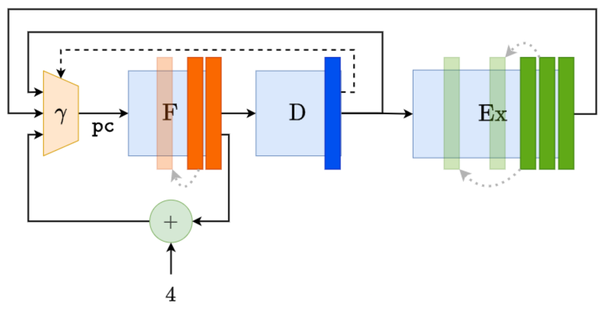




Exploring pipeline depths

- Intuitive pipeline depth specification
- Automated pipeline balancing using the backend HLS toolchain

7-stage pipeline



Conclusion

- We need speculation to synthesize processor cores [Nurvitadhi'2011, Josipovic'2019, Derrien'2020].
- Processor design from an ISS using **speculative pipelining** enables **fast iteration times** and **intuitive** design exploration.

References

[Derrien'2020] Derrien, S., Marty, T., Rokicki, S., and Yuki, T. (2020). *Toward speculative loop pipelining for high-level synthesis*. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 39(11):4229–4239.

[Josipovic'2019] Josipovic, L., Guerrieri, A., and Ienne, P. (2019). *Speculative dataflow circuits*. In Proceedings of the 2019 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, FPGA '19, page 162–171, ACM.

[Nurvitadhi'2011] Nurvitadhi, E., Hoe, J. C., Kam, T., and Lu, S.-L. L. (2011). *Automatic pipelining from transactional datapath specifications*. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 30(3):441–454.