Hardware/Software Co-Analysis for Worst Case Execution Time Bounds

Karlsruhe Institute of Technology

Chair for Embedded Systems

Can Joshua Lehmann (can.l@posteo.de), Lars Bauer, Hassan Nassar, Heba Khdr, Jörg Henkel

Worst Case Execution-Time (WCET)

Traditional WCET Tools [2, 3, 4, 7]

- Find upper bounds on the WCET of a program binary on a target processor
- Require a timing model of target processor

Problem

Finding WCET bounds in the presence of **Custom Instruction Set Extensions**

→ No timing model available

Our Solution

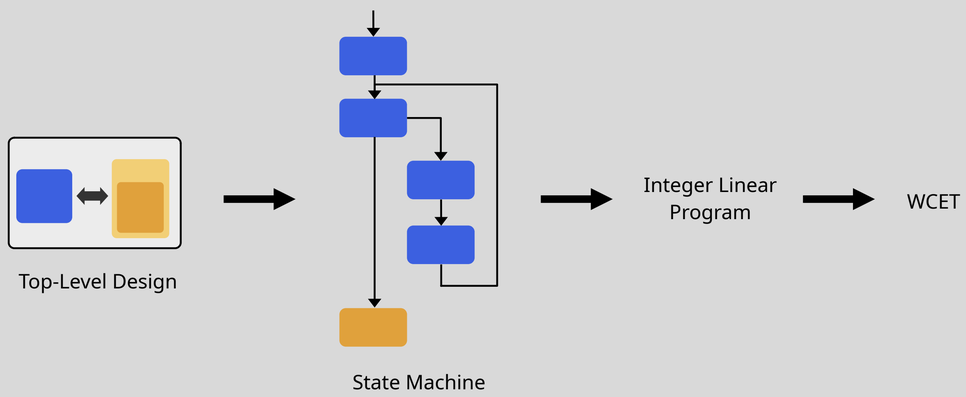
Find WCET based on hardware description of the target processor

→ No timing model required

→ Automatically handles custom instruction set extensions

Our Method

HW/SW Co-Analysis based Method

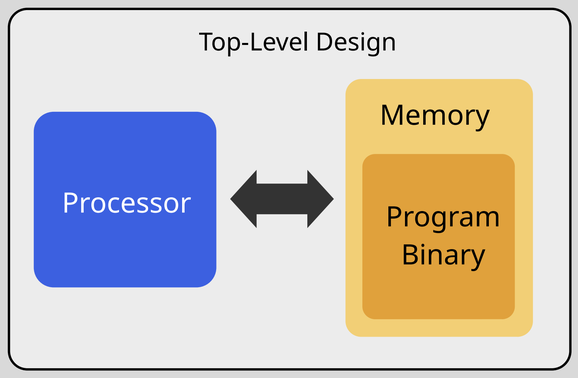


1. Instantiate processor and memory into combined top-level design
2. Reconstruct a state machine from top-level design
   - Large-scale structure is similar to control flow graph
   - Already contains microarchitectural states
     - Each transition corresponds to exactly one cycle
3. Apply implicit path enumeration technique [8] (IPET) to the state machine to encode the WCET problem as an integer linear program (ILP)
4. Solve ILP to obtain WCET

Hardware/Software Co-Analysis [5, 6]

Instantiate processor and memory in top-level design

- Memory contains application binary
- Analyze as single, combined design



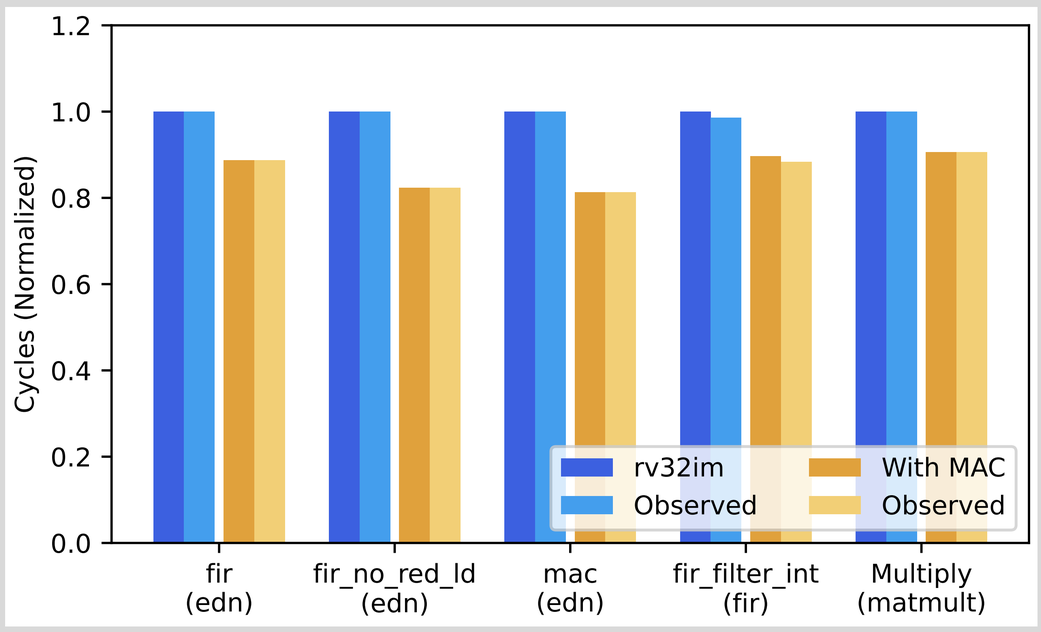
Symbolic Trajectory Evaluation based Approach

- Start simulation from reset state
- When unknown value (X) reaches the program counter (PC), a branch has been encountered.
  - Spawn new simulations for each branch target
- Merge simulation state with equivalent PC

Results and Evaluation

Setup

- Extend FemtoRV32 Individua RISC-V processor [9] with multiply accumulate (MAC) custom instruction from CORE-V Extension [11]
- Evaluate on Mälardalen benchmark set [10]
  - Benchmarks are modified to make use of custom MAC instruction
- Compare WCET bounds computed by our method for `rv32im` and `rv32im_Xcvmac`
- Compare to measured execution times of the benchmarks to evaluate accuracy of our method



Results

- Our method automatically and accruately models the behaviour of the custom instruction