The CASM language
- based on Abstract State Machine (ASM)
- statements (rules) produce update sets
- update sets are merged either parallel or sequential
- resulting update set applied when computation step concludes
- interleaving: state transformation - rule evaluation - state transformation - rule evaluation - ...
- efficient compilation
- symbolic execution (generating first-order logic predicates)

Model Verification
(using first-order logic)

The Problem: Are the pipelined instruction models combined with the pipeline and execution model coherent to the specification (instruction set models and its execution model)?
Solution: For each instruction both models are symbolically executed using the same initial state. A conjecture stating that the final states are equal is emitted. The fully-automated theorem prover vampire is used to perform the simulation proofs.

Execution Model
This rule defines one computation step of the CASM program. It is executed until the MIPS program triggers a trap.

\[
\text{rule run_program} \: \text{dumps (GPR, LO, HI)} \rightarrow \text{trace} =
\]
\[
\text{let branch = BRANCH in}
\]
\[
\text{let imm = PARG (addr, FV_IMM) in}
\]
\[
\text{let rt = PARG (addr, FV_RT) in}
\]
\[
\text{if rt != 0 then}
\]
\[
\text{GPR (rt) := BVand (32, GPR (rs), BVZeroExtend (imm, 16, 32))}
\]
\[
\text{if trapped then}
\]
\[
\text{if branch = undef then}
\]
\[
\text{PC := BRANCH}
\]
\[
\text{PC := PC + 4}
\]
\[
\text{print "program stopped (trapped)"}
\]
\[
\text{BRANCH := undef}
\]
\[
\text{program (self) := undef}
\]
\[
\text{endseqblock}
\]

Casino Synthesis
(compiling CASM to C++)
- Models compiled to C++
- ELF loader written in C++
- MIPS instruction decoder interfacing CASM written in C++
- MIPS assembler interfacing the host C library written in C++
- C++ library implementing arithmetic operations on bit vectors
- need to link MIPS program to simulator specific C library
- able to access host file system and perform terminal I/O

This work is partially supported by the Austrian Research Promotion Agency (FFG) under contract 827485, Correct Compilers for Correct Application Specific Processors and Catena DSP GmbH.

Andreas Krall <andi@complang.tuwien.ac.at>
Roland Lezuo <rlezuo@complang.tuwien.ac.at>
Vienna University of Technology
Institute of Computer Languages (E185)
1040 Vienna, Austria