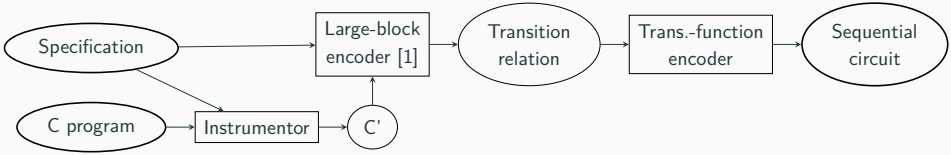
CPV: A Circuit-Based Program Verifier

Po-Chun Chien[1] and Nian-Ze Lee[2,1]

[1]LMU Munich · [2]National Taiwan University

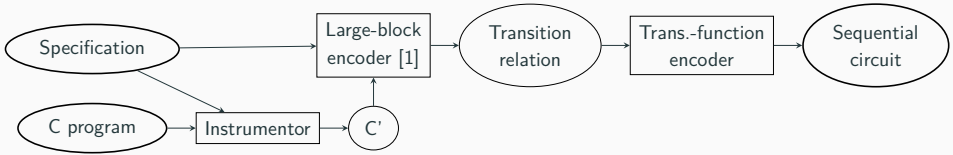
SV-COMP 2026 @ Turin, Italy

Verification Pipeline of CPV

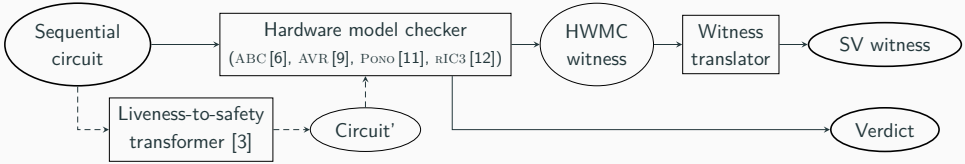


Frontend: Program instrumentation and encoding

Verification Pipeline of CPV



Frontend: Program instrumentation and encoding

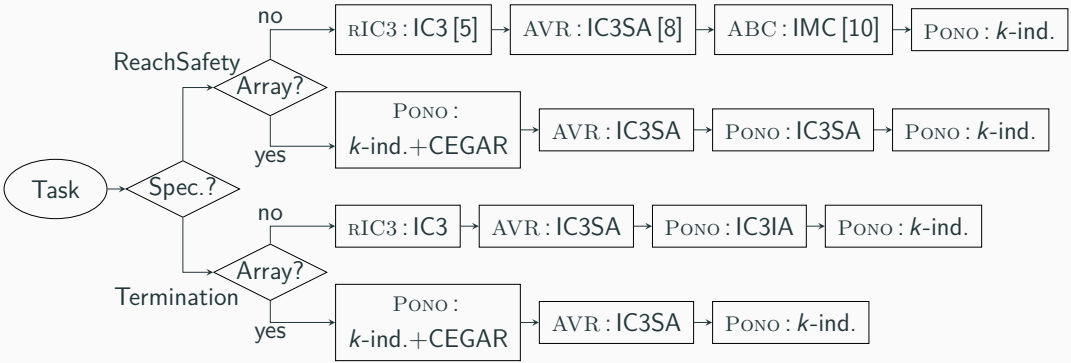


Backend: Model checking and witness translation

Enhancements for SV-COMP 2026

- Support Termination analysis (via liveness-to-safety reduction [3])
- Export violation witness v2
- Fix several issues in frontend task translator
- Integrate 2 new backend hardware model checkers: Pono [11] and rIC3 [12]
- Design a new strategy for SV-COMP

Strategy for SV-COMP 2026



CPV's Results at SV-COMP 2026: Highlights

C.ReachSafety

🥇 **1st:**

- Most #proofs overall (confirmed + unconfirmed)
- BitVectors, Hardware

🥈 **2nd:**

- 2nd most #solved overall (confirmed + unconfirmed)
- ECA, ProductLines, XCSP

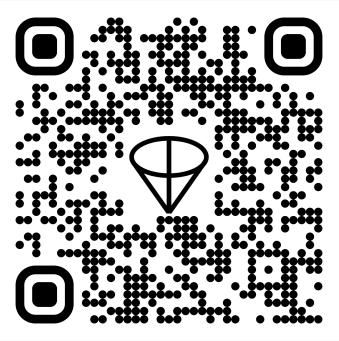
🥉 **3rd:** Arrays, Combinations, Hardness, Sequentialized

C.Termination

🥉 **3rd:** BitVectors

Conclusion

- With CPV [7], we
  - encode SV tasks as circuits
  - leverage powerful hardware model checkers as backend
- Strong performance in SV-COMP [2]
- Submitted translated BTOR2 circuits to HWMCC [4]
- Next steps:
  - Correctness witness translation
  - Extend frontend support



gitlab.com/
sosy-lab/software/cpv

References i

[1] Beyer, D., Cimatti, A., Griggio, A., Keremoglu, M.E., Sebastiani, R.: Software model checking via large-block encoding. In: Proc. FMCAD. pp. 25–32. IEEE (2009). https://doi.org/10.1109/FMCAD.2009.5351147

[2] Beyer, D., Strejček, J.: Evaluating software verifiers for C, Java, and SV-LIB (report on SV-COMP 2026). In: Proc. TACAS (2). LNCS 16506, Springer (2026)

[3] Biere, A., Artho, C., Schuppan, V.: Liveness checking as safety checking. In: Proc. FMICS. pp. 160–177. No. 2 in ENTSC 66, Elsevier (2002). https://doi.org/10.1016/S1571-0661(04)80410-9

[4] Biere, A., Froleyks, N., Preiner, M.: Hardware model checking competition 2025. In: Proc. FMCAD. p. 7. TU Wien Academic Press (2025). https://doi.org/10.34727/2025/isbn.978-3-85448-084-6_6

[5] Bradley, A.R.: SAT-based model checking without unrolling. In: Proc. VMCAI. pp. 70–87. LNCS 6538, Springer (2011). https://doi.org/10.1007/978-3-642-18275-4_7

References ii

[6] Brayton, R., Mishchenko, A.: ABC: An academic industrial-strength verification tool. In: Proc. CAV. pp. 24–40. LNCS 6174, Springer (2010). https://doi.org/10.1007/978-3-642-14295-6_5

[7] Chien, P.C., Lee, N.Z.: CPV: A circuit-based program verifier (competition contribution). In: Proc. TACAS (3). pp. 365–370. LNCS 14572, Springer (2024). https://doi.org/10.1007/978-3-031-57256-2_22

[8] Goel, A., Sakallah, K.: Model checking of Verilog RTL using IC3 with syntax-guided abstraction. In: Proc. NFM. pp. 166–185. Springer (2019). https://doi.org/10.1007/978-3-030-20652-9_11

[9] Goel, A., Sakallah, K.: AVR: Abstractly verifying reachability. In: Proc. TACAS. pp. 413–422. LNCS 12078, Springer (2020). https://doi.org/10.1007/978-3-030-45190-5_23

[10] McMillan, K.L.: Interpolation and SAT-based model checking. In: Proc. CAV. pp. 1–13. LNCS 2725, Springer (2003). https://doi.org/10.1007/978-3-540-45069-6_1

References iii

[11] Perez-Lopez, A.R., Chien, P.C., Lonsing, F., Archer, S., Irfan, A., Barrett, C.: Pono 2.0: A versatile SMT-based model checker for safety and liveness (long tool paper). In: Proc. FM. LNCS, Springer (2026)

[12] Su, Y., Yang, Q., Ci, Y., Bu, T., Huang, Z.: The rIC3 hardware model checker. In: Proc. CAV (1). pp. 185–199. LNCS 15931, Springer (2025). https://doi.org/10.1007/978-3-031-98668-0_9