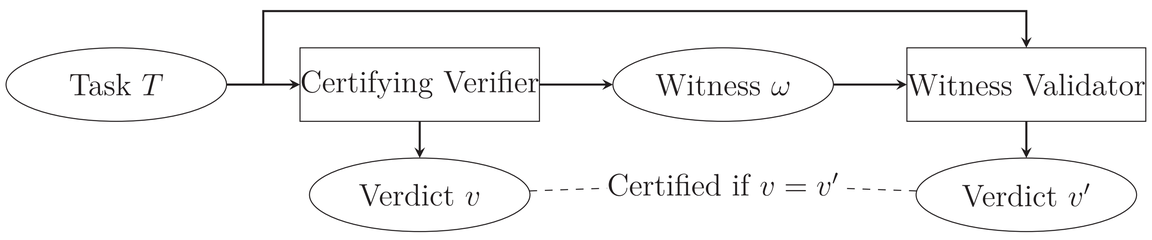
Btor2-Cert: A Certifying Hardware-Verification Framework Using Software Analyzers

Zsófia Ádám, Dirk Beyer, Po-Chun Chien, Nian-Ze Lee, and Nils Sirrenberg

adamzsofi@edu.bme.hu, nils.sirrenberg@campus.lmu.de,
{dirk.beyer,po-chun.chien,nian-ze.lee}@sosy.ifi.lmu.de

Presentation at TACAS 2024: 12:00, Thursday, April 11, Room: TBD

Certifying and Validating Verification



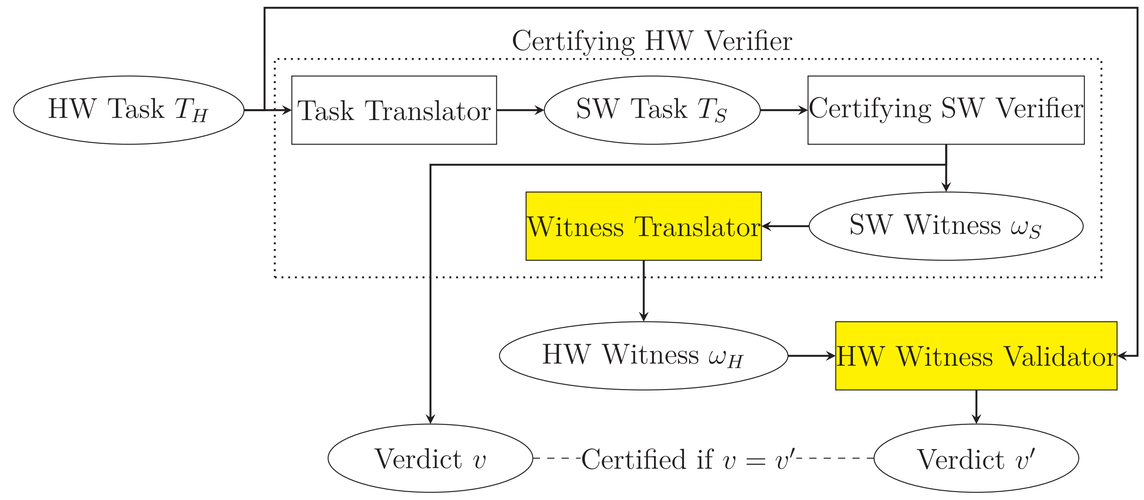
Our Motivation

- Explainable and trustworthy HW verification (HV)
- SW verification (SV) techniques for HW

Our Contributions

- A certifying HV framework using SV techniques
- A translator from SW witnesses to HW witnesses
- A witness validator for the Btor2 HW modeling language [6]
- Complementing HV with certified results from SV

Certifying HV Using Translation and SV



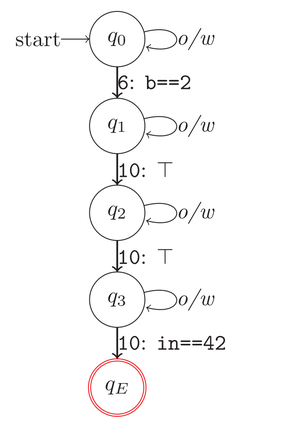
Btor2-Cert instantiates the framework with Btor2C [1] as frontend and SW verifiers that export GraphML witnesses [2] as backend.

HW-to-SW Translation via Btor2C [1]

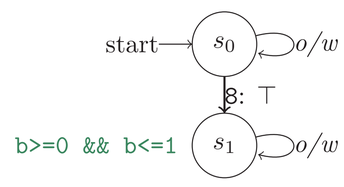
```
1 sort bitvec 8
2 sort bitvec 1
3 constd 1 42
4 constd 1 2
5 zero 1
6 state 1 ; a
7 state 1 ; b
8 input 1 ; in
9 init 1 6 4 ; a init to 2
10 init 1 7 5 ; b init to 0
11 eq 2 6 5 ; a == 0
12 eq 2 7 4 ; b == 2
13 eq 2 8 3 ; in == 42
14 and 2 11 12
15 and 2 13 14
16 bad 15
17 one 1
18 srl 1 6 17
19 xor 1 7 17
20 next 1 6 18
21 next 1 7 19
```

```
1  extern void abort(void);
2  extern unsigned char nondet_uchar();
3  void main() {
4    typedef unsigned char SORT_1;
5    SORT_1 a = nondet_uchar();
6    SORT_1 b = nondet_uchar();
7    a = 2;
8    b = 0;  // Omit for unsafe version
9    for (;;) {
10     SORT_1 in = nondet_uchar();
11     if (a == 0 && b == 2 && in == 42) {
12     ERROR:
13       abort();
14     }
15     a = a >> 1;
16     b = b ^ 1;
17   }
18 }
```

Witness Translation

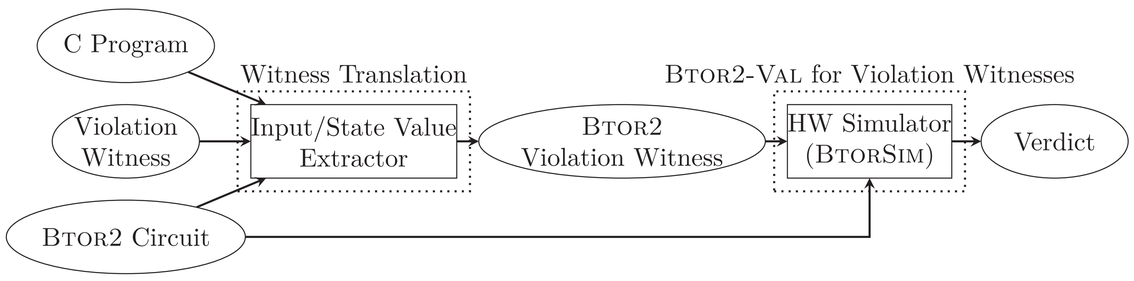



```
sat
b0
#0
1 00000010 ; b==2
@0
@1
@2
0 00101010 ; in==42
.
```

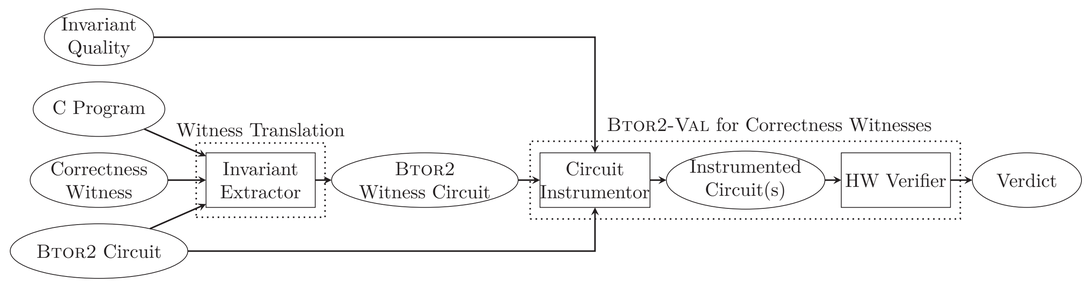



```
1 sort bitvec 8
2 sort bitvec 1
3 zero 1
4 one 1
5 input 1 ; state "b"
6 ugte 2 5 3 ; b >= 0
7 ulte 2 5 4 ; b <= 1
8 and 2 6 7
9 output 8
```

Violation Witness Validation



Correctness Witness Validation



Summary of Experimental Results

On 758 safe and 456 unsafe Btor2 verification tasks, Btor2-Cert achieved:

- Translation of all violation and 97 % correctness witnesses,
- Effective and efficient validation vs. compared validators, e.g., LIV [4] and CPA-w2t [3], and
- Certified bugs in 8 % of the unsafe tasks with CBMC [5] that HV overlooked

Invariant Quality

Three user-defined quality levels for invariants:

- Invariant (containing all reachable states)
- Safe invariant (implying safety property)
- Safe and inductive invariant

References

[1] Beyer, D., Chien, P.C., Lee, N.Z.: Bridging hardware and software analysis with Btor2C: A word-level-circuit-to-C translator. In: Proc. TACAS. pp. 1–21. LNCS 13994 (2023)
[2] Beyer, D., Dangl, M., Dietsch, D., Heizmann, M., Lemberger, T., Tautschnig, M.: Verification witnesses. ACM Trans. Softw. Eng. Methodol. **31**(4), 57:1–57:69 (2022)
[3] Beyer, D., Dangl, M., Lemberger, T., Tautschnig, M.: Tests from witnesses: Execution-based validation of verification results. In: Proc. TAP. pp. 3–23. LNCS 10889 (2018)
[4] Beyer, D., Spiessl, M.: LIV: A loop-invariant validation using straight-line programs. In: Proc. ASE. pp. 2074–2077 (2023)
[5] Clarke, E.M., Kröning, D., Lerda, F.: A tool for checking ANSI-C programs. In: Proc. TACAS. pp. 168–176. LNCS 2988 (2004)
[6] Niemetz, A., Preiner, M., Wolf, C., Biere, A.: Btor2, BtorMC, and Boolector 3.0. In: Proc. CAV. pp. 587–595. LNCS 10981 (2018)

Try Btor2-Cert!

Artifact DOI: 10.5281/zenodo.10548597