Verifying Bit-vector Invertibility Conditions in Coq (Extended Abstract)

Burak Ekici, Arjun Viswanathan, Yoni Zohar, Clark Barrett, Cesare Tinelli
Bit-vectors
Bit-vectors

• Bit-vectors: Fixed-width bit sequences
  10101
  $a \in \text{BV}_5$
Bit-vectors

• Bit-vectors: Fixed-width bit sequences
  10101
  $a \in \text{BV}_5$

• Bit-vector operations
  $a + b$
  $a <_u b$
  ...

Introduction

Bit-vectors have many applications:
• Hardware circuit analysis [Gupta et al., 1993]
• Bounded model checking [Armando et al., 2006]
• Symbolic execution [Cadar et al., 2006]
• ...
Introduction

• Many applications require **quantified** bit-vector formulas

• Some SMT solvers use quantifier-instantiation to solve quantified formulas

• *Invertibility conditions* are a useful meta-construct for a quantifier-instantiation technique [Niemetz et al., CAV 2018]
Invertibility Conditions

An *invertibility condition* for a variable \( x \) in a bit-vector literal

\[
\ell \left[ x, s, t \right]
\]

is a formula

\[
IC \left[ s, t \right]
\]

s.t. the following *invertibility equivalence* is valid in the theory of bit-vectors:

\[
\forall s. \forall t. IC[s, t] \iff \exists x. \ell[x, s, t]
\]

where \( s, t, x : BV_n \)
Invertibility Conditions: Example
Invertibility Conditions: Example

• Inversion of bit-vector addition is unconditional

\[ \exists x. x + s = t \iff T \]

The inverse is \( x = t - s \)
Invertibility Conditions: Example

• Inversion of bit-vector addition is unconditional

\[ \exists x. \ x + s = t \iff \top \]

The *inverse* is \( x = t - s \)

• Inversion of bit-wise conjunction is conditional

\[ \exists x. \ x \& s = t \iff t \& s = t \]
Motivation

This technique [Niemetz et al., CAV 2018] requires the equivalences to be true independent of bit-width.

Proofs of these equivalences are required for the soundness of the technique.

...and the solvers that use it.
Previous Work
Previous Work

[Niemetz et al., CAV 2018]
• generated 162 invertibility equivalences
• proved them using SMT-solvers for bit-widths up to 65
Previous Work

[Niemetz et al., CAV 2018]
• generated 162 invertibility equivalences
• proved them using SMT-solvers for bit-widths up to 65

[Niemetz et al., CADE 2019]
• encoded the equivalences in theories supported by SMT-solvers
• verified equivalences for parametric bit-widths
• approach succeeded on under 75% of the equivalences
Contributions
Contributions

1. Formalized a representative subset of the 162 invertibility equivalences in Coq
Contributions

1. Formalized a representative subset of the 162 invertibility equivalences in Coq

2. Extended a Coq bit-vector library to support these equivalences
Contributions

1. Formalized a representative subset of the 162 invertibility equivalences in Coq

2. Extended a Coq bit-vector library to support these equivalences

3. Proved 18 of them for arbitrary bit-width
### Result Summary

<table>
<thead>
<tr>
<th>$\ell[x]$</th>
<th>$=$</th>
<th>$\neq$</th>
<th>$&lt;_u$</th>
<th>$&gt;_u$</th>
<th>$\leq_u$</th>
<th>$\geq_u$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$-x \otimes t$</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
</tr>
<tr>
<td>$\sim x \otimes t$</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
</tr>
<tr>
<td>$x &amp; s \otimes t$</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
</tr>
<tr>
<td>$x \mid s \otimes t$</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
</tr>
<tr>
<td>$x \ll s \otimes t$</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
</tr>
<tr>
<td>$s \ll x \otimes t$</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
</tr>
<tr>
<td>$x \gg s \otimes t$</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✗</td>
</tr>
<tr>
<td>$s \gg x \otimes t$</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
</tr>
<tr>
<td>$x \gg a s \otimes t$</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
</tr>
<tr>
<td>$s \gg a x \otimes t$</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
</tr>
<tr>
<td>$x + s \otimes t$</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
</tr>
</tbody>
</table>
## Result Summary (SMT)

<table>
<thead>
<tr>
<th>( \ell[x] )</th>
<th>=</th>
<th>( \neq )</th>
<th>(&lt;_u)</th>
<th>(&gt;_u)</th>
<th>(\leq_u)</th>
<th>(\geq_u)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(-x \otimes t)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(~ x \otimes t)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(x &amp; s \otimes t)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(x</td>
<td>s \otimes t)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(x &lt;&lt; s \otimes t)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(s &lt;&lt; x \otimes t)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(x &gt;&gt; s \otimes t)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(s &gt;&gt; x \otimes t)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(x &gt;&gt;_a s \otimes t)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(s &gt;&gt;_a x \otimes t)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(x + s \otimes t)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>$\ell[x]$</td>
<td>$=$</td>
<td>$\neq$</td>
<td>$&lt;_{u}$</td>
<td>$&gt;_u$</td>
<td>$\leq_{u}$</td>
<td>$\geq_{u}$</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----</td>
<td>--------</td>
<td>---------</td>
<td>-------</td>
<td>-----------</td>
<td>-----------</td>
</tr>
<tr>
<td>$-x \otimes t$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>$\sim x \otimes t$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>$x &amp; s \otimes t$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>$x \mid s \otimes t$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>$x \ll s \otimes t$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>$s \ll x \otimes t$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>$x \gg s \otimes t$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>$s \gg x \otimes t$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>$x \gg_a s \otimes t$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>$s \gg_a x \otimes t$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>$x + s \otimes t$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>
## Result Summary (Both)

<table>
<thead>
<tr>
<th>( \ell[x] )</th>
<th>=</th>
<th>≠</th>
<th>(&lt;_u)</th>
<th>(&gt;_u)</th>
<th>(\leq_u)</th>
<th>(\geq_u)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(-x \otimes t)</td>
<td>✓✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(\sim x \otimes t)</td>
<td>✓✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(x &amp; s \otimes t)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(x</td>
<td>s \otimes t)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(x \ll s \otimes t)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(x \lll s \otimes t)</td>
<td>✓✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(x \gg s \otimes t)</td>
<td>✓✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(s \gg x \otimes t)</td>
<td>✓✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(x \ggg a s \otimes t)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(s \ggg a x \otimes t)</td>
<td>✓✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(x + s \otimes t)</td>
<td>✓✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

✓ Verified in Coq
✓ Verified in SMT
✓✓ Verified in Coq and SMT
✗ Verified in neither Coq nor SMT
Bit-vector Library
Bit-vector Library

• Used a bit-vector library originally developed for SMTCoq [Ekici et al., 2017]
• SMTCoq is a Coq plugin that uses external SMT solvers to complete proof goals
• Bit-vectors are represented as lists of Booleans
Bit-vector Library

- Used a bit-vector library originally developed for SMTCoq [Ekici et al., 2017]
- SMTCoq is a Coq plugin that uses external SMT solvers to complete proof goals
- Bit-vectors are represented as lists of Booleans

```
Modules

Dependently Typed Bit-vector

Types

bitvector : N -> Type

Functor: Raw2Bitvector

bitvector : Type
size : bitvector -> N
```
Bit-vector Representations

<table>
<thead>
<tr>
<th>Bit-vector Representation</th>
<th>SMTLib [Niemetz et al. 18]</th>
<th>Encoding [Niemetz et al. 19]</th>
<th>Coq Library</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Bit-vector of width n One sort for each n</td>
<td>Bit-vector of width n Translated to NIA and UF</td>
<td>Bit-vector of width n List of Booleans over 2 layers</td>
</tr>
<tr>
<td>Expressivity:</td>
<td>n cannot be symbolic</td>
<td>Allows quantification over n</td>
<td>Bit-vectors dependent over n</td>
</tr>
<tr>
<td>Verification:</td>
<td>Automatic proofs using SMT solvers</td>
<td>Automatic proofs using SMT solvers</td>
<td>Manual proofs in Coq</td>
</tr>
<tr>
<td>Results</td>
<td>All equivalences for n = 1 to 65</td>
<td>Verified ≈75% of equivalences</td>
<td>18 equivalences</td>
</tr>
</tbody>
</table>
Bit-vector Library

Basic signature (previous work):

+     addition
-     negation
•     multiplication
&     bit-wise conjunction
|     bit-wise disjunction
~     bit-wise negation
<<    logical left shift
>>    logical right shift

_EXTENDED SIGNATURE (this work):

≤      unsigned weak less than      <=    logical left shift (alt. def.)
≥      unsigned weak greater than   >=    logical right shift (alt. def.)
>>a    arithmetic right shift       >>a   arithmetic right shift (alt. def.)
Original Shift Definition

**Definition** shl_one_bit (a: list bool) : list bool :=
    match a with
    | [] => []
    | _ => false :: removelast a
end.

**Fixpoint** shl_n_bits (a: list bool) (n: nat): list bool :=
    match n with
    | 0 => a
    | S n' => shl_n_bits (shl_one_bit a) n'
end.

**Definition** shl_aux (a b: list bool): list bool :=
    shl_n_bits a (list2nat_be_a b).
Shift Redefined

**Definition** shl\_n\_bits\_a (a: list bool) (n: nat): list bool :=
if (n <? length a)\%nat then
mk\_list\_false n ++ firstn (length a - n) a
else
mk\_list\_false (length a).

**Definition** bv\_shl\_a (a b: bitvector): bitvector :=
if ((@size a) =? (@size b)) then
shl\_n\_bits\_a a (list2nat\_be\_a b)
else
nil.
Invertibility Conditions
Proofs
## Result Summary

<table>
<thead>
<tr>
<th>$\ell[x]$</th>
<th>$=$</th>
<th>$\neq$</th>
<th>$\leq_u$</th>
<th>$&lt;u$</th>
<th>$&gt;u$</th>
<th>$\leq_u$</th>
<th>$\geq_u$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$-x \bowtie t$</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
</tr>
<tr>
<td>$\sim x \bowtie t$</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
</tr>
<tr>
<td>$x &amp; s \bowtie t$</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
</tr>
<tr>
<td>$x \mid s \bowtie t$</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
</tr>
<tr>
<td>$x \ll s \bowtie t$</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
</tr>
<tr>
<td>$s \ll x \bowtie t$</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
</tr>
<tr>
<td>$x \gg s \bowtie t$</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
</tr>
<tr>
<td>$s \gg x \bowtie t$</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
</tr>
<tr>
<td>$x \gg_a s \bowtie t$</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
</tr>
<tr>
<td>$s \gg_a x \bowtie t$</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
</tr>
<tr>
<td>$x + s \bowtie t$</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
</tr>
</tbody>
</table>
# Result Summary

<table>
<thead>
<tr>
<th>$\ell[x]$</th>
<th>$=$</th>
<th>$\neq$</th>
<th>$\leq_u$</th>
<th>$&gt;u$</th>
<th>$\leq_u$</th>
<th>$\geq_u$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$-x \triangleq t$</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
</tr>
<tr>
<td>$\sim x \triangleq t$</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
</tr>
<tr>
<td>$x &amp; s \triangleq t$</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
</tr>
<tr>
<td>$x \mid s \triangleq t$</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
</tr>
<tr>
<td>$x \ll s \triangleq t$</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
</tr>
<tr>
<td>$s \ll x \triangleq t$</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
</tr>
<tr>
<td>$x \gg s \triangleq t$</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
</tr>
<tr>
<td>$s \gg x \triangleq t$</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
</tr>
<tr>
<td>$x \gg_a s \triangleq t$</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
</tr>
<tr>
<td>$s \gg_a x \triangleq t$</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
</tr>
<tr>
<td>$x + s \triangleq t$</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
</tr>
</tbody>
</table>

✓ Verified in Coq
✓ Verified in SMT
✓✓ Verified in Coq and SMT
✗ Verified in neither Coq nor SMT
# Result Summary

<table>
<thead>
<tr>
<th>( \ell[x] )</th>
<th>=</th>
<th>( \neq )</th>
<th>(&lt;_u)</th>
<th>(&gt;_u)</th>
<th>(\leq_u)</th>
<th>(\geq_u)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(-x \gg t)</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
</tr>
<tr>
<td>(\sim x \gg t)</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
</tr>
<tr>
<td>(x &amp; s \gg t)</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
</tr>
<tr>
<td>(x \mid s \gg t)</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
</tr>
<tr>
<td>(x \ll s \gg t)</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
</tr>
<tr>
<td>(s \ll x \gg t)</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
</tr>
<tr>
<td>(x \gg s \gg t)</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
</tr>
<tr>
<td>(s \gg x \gg t)</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
</tr>
<tr>
<td>(x \gg_a s \gg t)</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
</tr>
<tr>
<td>(s \gg_a x \gg t)</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
</tr>
<tr>
<td>(x + s \gg t)</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
<td>✓✓</td>
</tr>
</tbody>
</table>

\textbf{bvshr\_ugt\_rtl}: \(\forall n. \forall x, s, t: BV_n.\)  
\((x \gg s) <_u t \rightarrow t <_u (\sim s \gg s)\)

✓ Verified in Coq  
✓ Verified in SMT  
✓✓ Verified in Coq and SMT  
✗ Verified in neither Coq nor SMT
Challenge
Challenge

Consider bit-vectors $s$ where $k < l$.

For $l = 4$,

<table>
<thead>
<tr>
<th>$s$</th>
<th>$k$</th>
<th>$l - k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>0001</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>0010</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>0011</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>

$k = \text{toNat}(s)$

$l = \text{length}(s)$
Consider bit-vectors $s$ where $k < l$.

For $l = 4$,

<table>
<thead>
<tr>
<th>s</th>
<th>k</th>
<th>l − k</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>0001</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>0010</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>0011</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>

$k = \text{toNat}(s)$

$l = \text{length}(s)$
Challenge

Consider bit-vectors $s$ where $k < l$.

For $l = 4$, 

<table>
<thead>
<tr>
<th>s</th>
<th>k</th>
<th>l - k</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>0001</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>0010</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>0011</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>

More generally, 

$$\text{msb}_\text{zero}: \forall n. \forall s: BV_n. k < l \rightarrow s[(l - 1)...k] = [0...0]$$

$k = \text{toNat}(s)$

$l = \text{length}(s)$
Challenge

Consider bit-vectors $s$ where $k < l$.

For $l = 4$,

<table>
<thead>
<tr>
<th>$s$</th>
<th>$k$</th>
<th>$l - k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>0001</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>0010</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>0011</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>

More generally,

$$\text{msb_zero}: \forall n. \forall s : BV_n. k < l \rightarrow s[(l - 1)\ldots k] = [0\ldots0]$$

$k = \text{toNat}(s)$

$l = \text{length}(s)$

$$\text{bvshr\_ugt\_rtl}: \forall n. \forall x, s, t : BV_n. (x \gg s) <_u t \rightarrow t <_u (\sim s \gg s)$$

reduces to

$\text{msb\_zero}$
Proof Sketch of msb_zero

msb_zero: ∀s : BV_n. k < l → s[(l - 1)...k] = [0...0]

Proof Sketch:

\[ s: \begin{bmatrix} l - 1 & k & k - 1 & 0 \end{bmatrix} \]

For \( l = 4 \),

<table>
<thead>
<tr>
<th>s</th>
<th>k</th>
<th>l - k</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>0001</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>0010</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>0011</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>
Proof Sketch of msb_zero

\[ \forall s : \text{BV}_n. \ k < l \rightarrow s[(l - 1) \ldots k] = [0 \ldots 0] \]

Proof Sketch:
\[
k = \sum_{i=0}^{l-1} s[i] \cdot 2^i
\]

\[
s: \begin{bmatrix}
    l - 1 & k & k - 1 & 0 \\
    0 & \ldots & 0 & \ldots & \ldots & \ldots
\end{bmatrix}
\]

For \( l = 4 \),

<table>
<thead>
<tr>
<th>( s )</th>
<th>( k )</th>
<th>( l - k )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>0001</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>0010</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>0011</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>
Proof Sketch of msb_zero

\text{msb\_zero: } \forall s : BV_n. \ k < l \rightarrow s[(l - 1)...k] = [0...0]

Proof Sketch:
\[ k = \sum_{i=0}^{l-1} s[i] \cdot 2^i \]
\[ = s[l - 1] \cdot 2^{l-1} + ... + s[1] \cdot 2^1 + s[0] \cdot 2^0 \]
Proof Sketch of `msb_zero`

**msb_zero**: \( \forall s : BV_n. \ k < l \rightarrow s[(l - 1)...k] = [0...0] \)

Proof Sketch:
\[
k = \sum_{i=0}^{l-1} s[i] \cdot 2^i
\]
\[
= s[l - 1] \cdot 2^{l-1} + ... + s[1] \cdot 2^1 + s[0] \cdot 2^0
\]

But \( k < l \)

\[
s: \begin{bmatrix} l - 1 & k & k - 1 & 0 \\ 0 & ... & 0 & - & ... & - \end{bmatrix}
\]

For \( l = 4 \),

<table>
<thead>
<tr>
<th>( s )</th>
<th>( k )</th>
<th>( l - k )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>0001</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>0010</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>0011</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>
Proof Sketch of msb_zero

\textbf{msb\_zero:} \quad \forall s : BV_n. \ k < l \rightarrow s[(l - 1)\ldots k] = [0\ldots0]

Proof Sketch:
\[ k = \sum_{i=0}^{l-1} s[i] \cdot 2^i \]

\[ = s[l - 1] \cdot 2^{l-1} + \ldots + s[1] \cdot 2^1 + s[0] \cdot 2^0 \]

But \( k < l \)
\[ = s[l - 1] \cdot 2^{l-1} + \ldots + s[k] \cdot 2^k + s[k - 1] \cdot 2^{k-1} + \ldots + s[1] \cdot 2^1 + s[0] \cdot 2^0 \]

\begin{align*}
\text{s:} & \quad \begin{bmatrix} l - 1 & k & k - 1 & 0 \end{bmatrix} \\
\text{For } l = 4, & \\
\begin{array}{|c|c|c|}
\hline
\text{s} & \text{k} & l - \text{k} \\
\hline
0000 & 0 & 4 \\
0001 & 1 & 3 \\
0010 & 2 & 2 \\
0011 & 3 & 1 \\
\hline
\end{array}
\end{align*}
Proof Sketch of msb_zero

$\text{msb}_\text{zero}: \forall s : BV_n. k < l \rightarrow s[(l - 1)\ldots k] = [0\ldots0]$

Proof Sketch:

$$k = \sum_{i=0}^{l-1} s[i] \cdot 2^i$$

$$= s[l - 1] \cdot 2^{l-1} + \ldots + s[1] \cdot 2^1 + s[0] \cdot 2^0$$

But $k < l$

$$= s[l - 1] \cdot 2^{l-1} + \ldots + s[k] \cdot 2^k + s[k - 1] \cdot 2^{k-1} + \ldots + s[1] \cdot 2^1 + s[0] \cdot 2^0$$

But $k < 2^k < 2^{k+1} < \ldots < 2^{l-1}$

$$s: \begin{bmatrix} l - 1 & k & k - 1 & 0 \\ 0 & \ldots & 0 & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ \end{bmatrix}$$

For $l = 4$,

<table>
<thead>
<tr>
<th>s</th>
<th>k</th>
<th>l - k</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>0001</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>0010</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>0011</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>s</th>
<th>k</th>
<th>l - k</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>0001</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>0010</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>0011</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>
Proof Sketch of msb_zero

msb_zero: \( \forall s : BV_n.\ k < l \rightarrow s[(l-1)...k] = [0...0] \)

Proof Sketch:

\[
k = \sum_{i=0}^{l-1} s[i] \cdot 2^i
\]

\[
= s[l-1] \cdot 2^{l-1} + \ldots + s[1] \cdot 2^1 + s[0] \cdot 2^0
\]

But \( k < l \)

\[
= s[l-1] \cdot 2^{l-1} + \ldots + s[k] \cdot 2^k + s[k-1] \cdot 2^{k-1} + \ldots + s[1] \cdot 2^1 + s[0] \cdot 2^0
\]

But \( k < 2^k < 2^{k+1} < \ldots < 2^{l-1} \)

Thus, the coefficients of \( 2^k, \ldots, 2^{l-1} \) are 0.
Conclusion

• [Niemetz et al., CAV 2018] presented 162 invertibility conditions

• [Niemetz et al., CADE 2019] verified ≈75% of the equivalences using an encoding

• We complemented [Niemetz et al., CADE 2019] in proving all but one invertibility equivalences from 66 of them

• We did this in the Coq proof assistant

• We extended the Coq bit-vector library for SMTCoq to do this
Future Work

• Integrate the extended bit-vector library into SMTCoq

• Prove the remaining invertibility equivalences

• Extend the library with $\div \% \leq_s \geq_s$

• Refactor the library for improved readability and modularity

• Consider using MathComp library for future proofs
References


Useful Lemmas

**Lemma** firstn_all: \( \forall (l : \text{list} \ A), \text{firstn} \ (\text{length} \ l) \ l = l. \)

**Lemma** firstn_length_le: \( \forall (l : \text{list} \ A) (n : \text{nat}), n \leq \text{length} \ l \rightarrow \text{length} \ (\text{firstn} \ n \ l) = n. \)

**Lemma** firstn_length: \( \forall (n : \text{nat}) (l : \text{list} \ A), \text{length} \ (\text{firstn} \ n \ l) = \text{min} \ n \ (\text{length} \ l). \)

**Theorem** app_nil_r: \( \forall (l : \text{list} \ A), l \succeq [] = l. \)

**Lemma** app_length: \( \forall (l \ l' : \text{list} \ A), \text{length} \ (l \succeq l') = \text{length} \ l + \text{length} \ l'. \)