Signed Primitive Integers in Coq

Ana Borges, Raül Espejo and Joost J. Joosten

Summary

Signed primitive integers were added to the Coq kernel alongside the previously existing unsigned primitive integers.

Problem

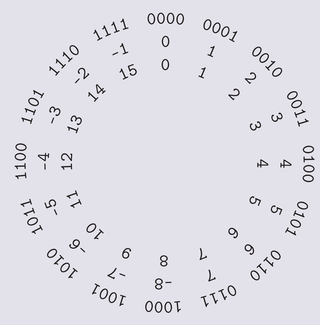
In mathematics it is useful to think of natural numbers as being either 0 or the successor S of some other natural number. *e.g.*, 3 is represented as $S\ (S\ (S\ 0))$. This definition is ideal for inductive proofs.

For computers, it is useful to think of natural numbers as their binary representation, consisting of 0s and 1s strings. *e.g.*, 3 can be represented as 11. This definition allows for efficient memory management and computation.

Coq provides the type `nat` for inductive proofs and the type `Uint63.int` of unsigned primitive integers for efficient memory management.

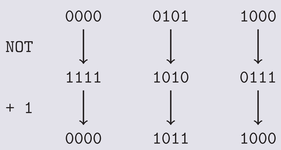
However, until recently there was no signed counterpart of `Uint63.int`. This contribution provides the signed version.

Successor



Cyclic behaviour of `+1` operator in 4 bits long integers, along with their's two's complement signed and unsigned interpretation.

Examples



The two's complement algorithm transforming 4 bits long integers 0, 5 and -8.

The minimal integer, -8, is invariant to two's complement.

Two's complement

A string of 0s and 1s can be interpreted either as an unsigned or a signed integer. In the signed interpretation with two's complement, the most significant bit is seen as a sign (0 for positive and 1 for negative). Thus, 63 bits are enough to express unsigned integers up to $2^{63} - 1$, and signed integers between -2^{62} and $2^{62} - 1$.

When interpreting binary strings with two's complement, some operations such as addition and multiplication work exactly the same under the unsigned and signed interpretations. However, some other operations such as division and comparison work differently.

Solution

The new type `Sint63.int` was added to Coq. This type is a two's complement interpretation of `Uint63.int` (previously named `Int63.int`), meaning that operations such as addition and multiplication are repurposed.

The new contributions are:

- Printing and parsing of binary strings as signed primitive integers.
- Primitive definitions of signed less-than-or-equal, signed less-than, signed division, signed remainder, and arithmetic shift right at the level of the Coq kernel.
- A Coq library `Sint63` with theory on signed primitive integers, including all the operations repurposed from `Uint63` as well as the newly defined ones.

References

Ana de Almeida Borges (FV), Guillaume Melquiond (INRIA) and Pierre Roux (ONERA) *Int63.Sint63* 2021:

`https://coq.inria.fr/library/Coq.Numbers.Cyclic.Int63.Sint63.html`

`http://formalvindications.com/post/primitive-machine-signed-integers/`

New code can be found in:

`https://github.com/coq/coq/pull/13559`

Older unsigned code can be found in:

`https://coq.github.io/doc/master/stdlib/Coq.Numbers.Cyclic.Int63.PrimInt63.html`

`https://coq.inria.fr/library/Coq.Numbers.Cyclic.Int63.Uint63.html`

Proyecto RTC-2017-6740-7 financiado por MCIN/AEI/10.13039/501100011033 y por FEDER.