Funcons-beta: Bits *

The PLanCompS Project

`Bits.cbs` | PLAIN | PRETTY

OUTLINE

Bits and bit vectors
Bits
Bit vectors

Bits and bit vectors

[*Type* bits
Datatype bit-vectors
Funcon bit-vector
Type bytes
Alias octets
Funcon bit-vector-not
Funcon bit-vector-and
Funcon bit-vector-or
Funcon bit-vector-xor
Funcon bit-vector-shift-left
Funcon bit-vector-logical-shift-right
Funcon bit-vector-arithmetic-shift-right
Funcon integer-to-bit-vector
Funcon bit-vector-to-integer
Funcon bit-vector-to-natural
Funcon unsigned-bit-vector-maximum
Funcon signed-bit-vector-maximum
Funcon signed-bit-vector-minimum
Funcon is-in-signed-bit-vector
Funcon is-in-unsigned-bit-vector]

Bits

Type bits ⇝ booleans

false represents the absence of a bit, true its presence.

*Suggestions for improvement: `plancomps@gmail.com`.
Reports of issues: `https://github.com/plancomps/CBS-beta/issues`.

Bit vectors

Datatype bit-vectors(N : natural-numbers) ::= bit-vector(_ : bits^N)

Type bytes $\rightsquigarrow$ bit-vectors(8)

Alias octets = bytes

Meta-variables BT <: bit-vectors(_)

Built-in Funcon bit-vector-not(_ : BT) : $\Rightarrow BT$

Built-in Funcon bit-vector-and(_ : BT, _ : BT) : $\Rightarrow BT$

Built-in Funcon bit-vector-or(_ : BT, _ : BT) : $\Rightarrow BT$

Built-in Funcon bit-vector-xor(_ : BT, _ : BT) : $\Rightarrow BT$

The above four funcons are the natural extensions of funcons from booleans to bit-vectors(N) of the same length.

Built-in Funcon bit-vector-shift-left(_ : BT, _ : natural-numbers) : BT

Built-in Funcon bit-vector-logical-shift-right(_ : BT, _ : natural-numbers) : BT

Built-in Funcon bit-vector-arithmetic-shift-right(_ : BT, _ : natural-numbers) : BT

Built-in Funcon integer-to-bit-vector(_ : integers, N : natural-numbers) : bit-vectors(N)

integer-to-bit-vector(M, N) converts an integer M to a bit-vector of length N, using Two's Complement representation. If the integer is out of range of the representation, it will wrap around (modulo 2^N).

Built-in Funcon bit-vector-to-integer(_ : BT) : $\Rightarrow$ integers

bit-vector-to-integer(B) interprets a bit-vector BV as an integer in Two's Complement representation.

Built-in Funcon bit-vector-to-natural(_ : BT) : $\Rightarrow$ natural-numbers

bit-vector-to-natural(BV) interprets a bit-vector BV as a natural number in unsigned representation.

Funcon unsigned-bit-vector-maximum(N : natural-numbers) : $\Rightarrow$ natural-numbers
$\rightsquigarrow$ integer-subtract(integer-power(2, N), 1)

Funcon signed-bit-vector-maximum(N : natural-numbers) : $\Rightarrow$ integers
$\rightsquigarrow$ integer-subtract(integer-power(2, integer-subtract(N, 1)), 1)

Funcon signed-bit-vector-minimum(N : natural-numbers) : $\Rightarrow$ integers
$\rightsquigarrow$ integer-negate(integer-power(2, integer-subtract(N, 1)))

Funcon is-in-signed-bit-vector(M : integers, N : natural-numbers) : ⇒ booleans

⇝ and(

integer-is-less-or-equal(M, signed-bit-vector-maximum(N)),

integer-is-greater-or-equal(M, signed-bit-vector-minimum(N)))

Funcon is-in-unsigned-bit-vector(M : integers, N : natural-numbers) : ⇒ booleans

⇝ and(

integer-is-less-or-equal(M, unsigned-bit-vector-maximum(N)),

integer-is-greater-or-equal(M, 0))