

SystemC - Data Types (06A)

SystemC

Copyright (c) 2012 Young W. Lim.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

Please send corrections (or suggestions) to youngwlim@hotmail.com.

This document was produced by using OpenOffice and Octave.

Based on the following original work

- [1] Aleksandar Milenkovic, 2002
CPE 626 The SystemC Language – VHDL, Verilog Designer's Guide
<http://www.ece.uah.edu/~milenska/ce626-02S/lectures/cpe626-SystemC-L2.ppt>
- [2] Alexander de Graaf, EEMCS/ME/CAS, 2010
SystemC: an overview ET 4351
ens.ewi.tudelft.nl/Education/courses/et4351/SystemC-2010v1.pdf
- [3] Joachim Gerlach, 2001
System-on-Chip Design with System of Computer Engineering
<http://www2.cs.uni-paderborn.de/cs/ag-hardt/Forschung/Data/SystemC-Tutorial.pdf>
- [4] Martino Ruggiero, 2008
SystemC
polimage.polito.it/~lavagno/codes/SystemC_Lezione.pdf
- [5] Deepak Kumar Tal, 1998-2012
SystemC Tutorial
<http://www.asic-world.com/systemc/index.html>

SystemC Data Types

Type	Description
sc_logic	Simple bit with 4 values(0/1/X/Z)
sc_int	Signed Integer from 1-64 bits
sc_uint	Unsigned Integer from 1-64 bits
sc_bigint	Arbitrary size signed integer
sc_biguint	Arbitrary size unsigned integer
sc_bv	Arbitrary size 2-values vector
sc_lv	Arbitrary size 4-values vector
sc_fixed	templated signed fixed point
sc_ufixed	templated unsigned fixed point
sc_fix	untemplated signed fixed point
sc_ufix	untemplated unsigned fixed point

Examples

- **bool** *2 value single bit type [0 or 1]*
`bool A, B;`
`sc_in<bool> input`
;
- **sc_logic** *4 value single bit type [0, 1, X or Z]*
`sc_logic C, D;`
`sc_out<sc_logic> E;`
- **sc_int** *[1 to 64]-bit signed integer type*
`sc_int<16> x, y;`
`sc_out<sc_int<16>> z;`
- **sc_time** *time (units: SC_PS, SC_NS, SC_MS etc.)*
`sc_time t1(10, SC_NS)`

Fixed Point Types (1)

Templated

static arguments <>

Untemplated

non-static arguments ()

Finite Precision

signed

sc_fixed

sc_fix

unsigned

sc_ufixed

sc_ufix

Limited Precision

_fast

signed

sc_fixed_fast

sc_fix_fast

unsigned

sc_ufixed_fast

sc_ufix_fast

Fixed Point Types (2)

Templated static arguments - can be know in compile time

```
sc_fixed < wl, iwl, q_mode, o_mode, n_bits > var_name (init_val);
```

```
sc_fixed_fast < wl, iwl, q_mode, o_mode, n_bits > var_name (init_val);
```

Untemplated non-static arguments - can be configured during run time

```
sc_fix var_name ( wl, iwl, q_mode, o_mode, n_bits );
```

```
sc_fix_fast var_name ( wl, iwl, q_mode, o_mode, n_bits );
```

set_fxtype_context useful for an array of the _fix types
can set default parameters

Fixed Point Parameters

```
sc_fixed < wl, iwl, q_mode, o_mode, n_bits > var_name (init_val);
```

```
sc_fix var_name ( wl, iwl, q_mode, o_mode, n_bits );
```

wl	- total number of bits
iwl	- number of integer bits
q_mode	- quantization mode
o_mode	- overflow_mode
n_bits	- number of bits for overflow mode

optional
for **_fixed**

optional
for **_fix**

q_mode	- quantization mode
SC_RND	Round
SC_RND_ZERO	Round towards zero
SC_RND_MIN_INF	Round towards minus infinity
SC_RND_INF	Round towards infinity
SC_RND_CONV	Convergent rounding
SC_TRN	Truncate
SC_TRN_ZERO	Truncate towards zero

o_mode	- overflow_mode
SC_SAT	Saturate
SC_SAT_ZERO	Saturate to zero
SC_SAT_SYM	Saturate symmetrically
SC_WRAP	Wraparound
SC_WRAP_SYM	Wraparound symmetrically

Fast Fixed-point Data Types

Arbitrary Precision vs. Simulation Speed

Achieving Faster Speed

- Use `double` as underlying data type
- Mantissa limited to 53 bits
- Range limited to that of `double`

Fast Fixed-Point Types

- `sc_fixed_fast`, `sc_ufixed_fast`
- `sc_fix_fast`, `sc_ufix_fast`

Exactly the same declaration format and usage as before

All fixed-point data types, can be mixed freely

SC_FIXED Example (1)

```
sc_fixed< wl, iwl, q_mode, o_mode, n_bits > var_name (init_val);
```

```
sc_fixed< 8, 4 > my_var (-1.75);
```

$$(1.75)_{10} = (0001.1100)_2$$

The diagram shows the binary representation (0001.1100)₂ of the decimal value 1.75. A bracket under the entire string '0001.1100' is labeled with the number 8, representing the total number of bits. A second bracket under the integer part '0001' is labeled with the number 4, representing the number of integer bits.

wl = 8 - total number of bits
iwl = 4 - number of integer bits

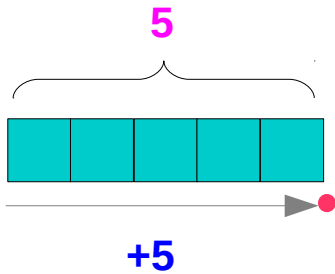
1's complement of (0001.1100)₂ = (1110.0011)₂

2's complement of (0001.1100)₂ = (1110.0100)₂

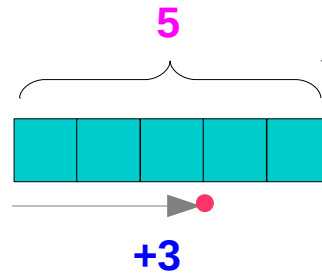
SC_FIXED Example (2)

`sc_fixed< wl, iwl, q_mode, o_mode, n_bits > var_name (init_val);`

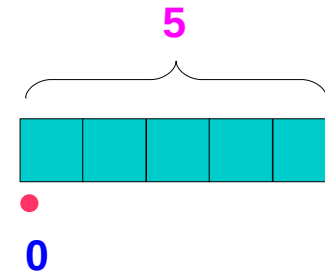
`sc_fixed< 5, 5 > A1;`



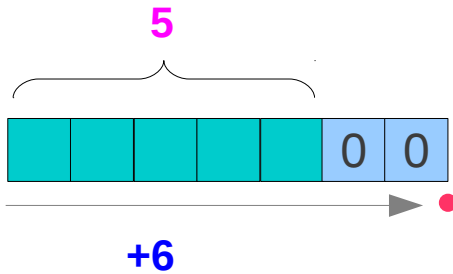
`sc_fixed< 5, 3 > A2;`



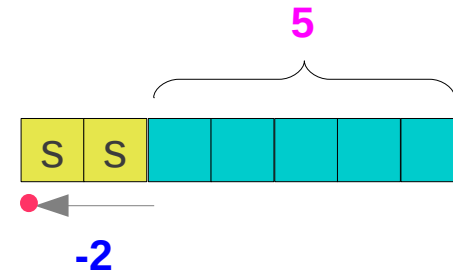
`sc_fixed< 5, 0 > A3;`



`sc_fixed< 5, 7 > A4;`



`sc_fixed< 5, -2 > A5;`



References

- [1] Aleksandar Milenkovic, 2002
CPE 626 The SystemC Language – VHDL, Verilog Designer’s Guide
<http://www.ece.uah.edu/~milenska/ce626-02S/lectures/cpe626-SystemC-L2.ppt>

- [2] Alexander de Graaf, EEMCS/ME/CAS, 2010
SystemC: an overview ET 4351
ens.ewi.tudelft.nl/Education/courses/et4351/SystemC-2010v1.pdf

- [3] Joachim Gerlach, 2001
System-on-Chip Design with System of Computer Engineering
<http://www2.cs.uni-paderborn.de/cs/ag-hardt/Forschung/Data/SystemC-Tutorial.pdf>

- [4] Martino Ruggiero, 2008
SystemC
polimage.polito.it/~lavagno/codes/SystemC_Lezione.pdf

- [5] Deepak Kumar Tal, 1998-2012
SystemC Tutorial
<http://www.asic-world.com/systemc/index.html>