NUMERIC TYPES FOR SYNTHESIS

Alex Zamfirescu

Vantage Analysis Systems Inc.
42808 Christy St. Suite 200
Fremont, CA. 94538
Phone: (510) 659 0901

Summary

The effort for developing a standard VHDL library containing packages for synthesis has led to progress in the following areas:

1. Interpretation of logic types
2. Manipulating numeric types
3. Representing design constraints
4. Identifying other data needed by synthesis

The talk will present the two packages implementing a proposed solution for handling in a standard way numeric types in synthesis and simulation (item 2 above).

Definitions for overloaded arithmetic operations for std.standard.bit_vector and types defined in ieee.std_logic_1164 package are introduced, the rationales for all major decisions are explained, and the body implementation alternatives are discussed.

Features that will benefit from the proposed changes in the VHDL '92 new standard are outlined.

Finally the users will learn about how a standard for numeric types in synthesis and simulation will benefit them, and how they can get involved and influence the process.
NUMERIC TYPES FOR SYNTHESIS

October 20, 1992

Alex Zamfirescu

Vantage Analysis Systems, Inc.
42808 Christy St. Suite 200
Fremont, CA 94538
USA
(510) 659-0901
anz@vas.com

(C) 1992 Vantage Analysis Systems, Inc.

OUTLINE

The VHDL '92 requirements analysis phase left SSIG a clear objective

Numeric types effort covers RTL level "synthesizable" types, arithmetic operations (like "+", "-", "*", "/", abs, mod, rem), comparison operators (like "<", "<=", ">", ">=", eq) and conversions between types

Today presenting
- main decisions taken so far
- a view of the synthesis library proposal
- future plans

Acknowledgements: I would like to thank all SSIG members for their participation and inspired ideas. Special recognition is due to Ken Scott, Dave Ackley and Jean-Michel Bergé for their contributions

72
SSIG NUMERIC TYPES INITIAL REQUIREMENTS

Consensus reached via E-mail

1. Unconstrained bit-arrays *types* representing *signed* and *unsigned* integers
2. Signed should be 2's complement
3. Enable X propagation
4. *No need* for hex, floating point (provide for later addition), resource allocation
5. Support IEEE standard logic type (no subtypes)
6. *Possible* multiple packages, prototype using 1987 VHDL, vector to/from integer

SOME INTERESTING DECISIONS

- 'LEFT', 'RIGHT' versus 'HIGH', 'LOW' for the order msb, lsb
  - VHDL array equality is defined based on 'LEFT' – 'RIGHT' order
  - S'LEFT' is the msb for any array object of subtype S

- Same size versus fill or extend
  - use different sizes when possible

- Field size constraints for procedures
  - for "+" and "-" result size is the largest operand size
  - for "*" result size is the sum of the operand sizes
  - for "/" result size is the dividend size

- Strong, weak, or as computed result values
  - only '0', '1', 'X', 'U' as elements of the result
**HOW X's INTERACT**

**X PROPAGATION**

\[
\begin{array}{c|c|c|c|c|c|c|c|c|c} 
0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 
0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 
\hline 
\end{array} \quad + \\
\begin{array}{c|c|c|c|c|c|c|c|c|c} 
0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 
0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 
\hline 
\end{array} \\

CASE A \quad \text{XXXXXXX010} \\
CASE B \quad \text{XXXXX0XXXXX} \\
CASE C \quad \text{XX1XX010} \\
\wedge \\

\rightarrow \text{Support C} \\
\rightarrow \text{Multiplication X propagation based on addition propagation}

---

**SIX ARRAY TYPES**

Two symmetric packages

```vhdl
package NUMERIC_BIT is
    type UNSIGNED_BV is array (NATURAL range <>) of BIT;
type COMP2_BV is array (NATURAL range <>) of BIT;
...
library IEEE;
use STD_LOGIC_1164.all;
package NUMERIC_STD is
    type UNSIGNED_LV is array (NATURAL range <>) of STD_ULOGIC;
type COMP2_LV is array (NATURAL range <>) of STD_ULOGIC;
```

74
A DESIRED PROPERTY

Corresponding operations are defined in both packages so that

to_std_logic ( NUMERIC_BIT.any_op ( array_1, array_2 ) ) =
NUMERIC_STD.corresponding_op ( to_std_logic (array_1),
to_std_logic (array_2) )

NUMERIC_BIT

| type T |
| a_1 op | z |
| a_2 |

NUMERIC_STD

| type image_T |
| image_op |
| y |
| image_a_1 |
| image_a_2 |

y is the image of z

SYNTHESIS LIBRARY—AT-A-GLANCE

package NUMERIC_BIT

BV

BV

BV

BV

BV

BV

BV

BITVECTOR

UNSIGNED_BV

COMP2_BV

package NUMERIC_STD

STD_LOGIC_VECTOR

UNSIGNED_LV

COMP2_LV

function

procedure
NEW TOOL CHALLENGES

SYNTHESIS

- Support as much as possible
- Selection criteria that improve quality

SIMULATION

- Special treatment expected
  - speed
    - acceleration using foreign interfaces
    - pre-compiled library
  - friendly interface
    - to the library
    - run time special support

FORMAL TECHNIQUES

- Take advantage of the relative reduced problem complexity
- Formally validated procedures

---

VHDL '92 FEATURES

<table>
<thead>
<tr>
<th>LCS #</th>
<th>TOPIC</th>
<th>IMPACT</th>
</tr>
</thead>
<tbody>
<tr>
<td>05</td>
<td>private types ..................</td>
<td>possible use to enable optimizations</td>
</tr>
<tr>
<td>08</td>
<td>new aliasing...................</td>
<td>use aliases for procedures in allocations</td>
</tr>
<tr>
<td>10</td>
<td>xnor ................................</td>
<td>affects 1164 directly</td>
</tr>
<tr>
<td>14</td>
<td>language--d att.....use 'ASCENDING, 'VALUE, 'IMAGE</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>logging messages..............</td>
<td>use in implementation and testing</td>
</tr>
<tr>
<td>18</td>
<td>concatenation.................</td>
<td>affecting by definition shifts and rotate</td>
</tr>
<tr>
<td>22</td>
<td>access to pred oper....(uses LCS 08)</td>
<td>use &quot;=&quot; instead of eq</td>
</tr>
<tr>
<td>24</td>
<td>shift and rotate.............</td>
<td>default for bit arrays, overload for std</td>
</tr>
<tr>
<td>26</td>
<td>foreign interface............</td>
<td>enable optimizations, validation needed</td>
</tr>
<tr>
<td>41</td>
<td>run--t constraint check.........</td>
<td>increased robustness</td>
</tr>
<tr>
<td>46</td>
<td>global variables................</td>
<td>possible store flags</td>
</tr>
</tbody>
</table>
WHAT'S NEXT

Prototype package implementations
Test, validate usability, check portability issues
Propose it as a standard and follow legal IEEE procedures
Possible split and ballot before other outcome from SSIG
Get involved!

CONCLUSIONS

- A strong start backed by a big momentum
- SSIG has the required expertise – no new inventions needed
- VHDL '92 has little impact on the proposed library
- Much work still ahead
- All pertinent input taken into account – consensus so far
- Another level where industry-wide cooperation and competition are orthogonal