A problem

Write a function named Is_X that:
- has a std_logic_vector as an input,
- returns a Boolean,
- And returns a TRUE if the vector contains a 'U', 'X', 'Z', 'W' or '-'; otherwise, it returns FALSE.

VHDL in Motion

Chapter 4
Signal Drivers
and Delay Models
pp. 135-150, 205-212

Review

- Last time: Subprograms
  - Functions
    - Return a single value, cannot modify inputs
    - No side effects
  - Procedures
    - Can return multiple values, modify inputs
    - May have side effects

Covered in This Lesson

- Signal Drivers
- Resolution Functions
- Delay Models
  - Inertial
  - Transport
- Waveform Updating Algorithms

Signal Drivers

- For every signal assigned a value within a process, a signal driver is created within the simulator.
- Think of this as an output of a logic gate:

Signal Drivers

- What happens when two or more signal drivers drive the same wire?

\[
\begin{align*}
a & \quad \quad \quad \quad \quad \quad \quad D \quad Q \\
b & \quad \quad \quad \quad \quad \quad \quad ?
\end{align*}
\]

When \( a \neq b \)?
Signal Drivers

- Often, we will deliberately connect outputs of gates together:

Tristate drivers: Each driver produces one of three values: '1', '0', or 'Z' (maybe logic type should also include 'X' and 'z')

CASE 1

Example of Multiple Signal Drivers in VHDL

- For every signal assigned a value within a process, a signal driver is created within the simulator:

A <= '1';
Fred: process (A) begin
BUS <= '0', '1' after 10 ns, 'Z' after 20 ns;
end process Fred;
Sam: process (A) begin
BUS <= '0', '1' after 5 ns, '0' after 15 ns;
end process Sam;

Another Example of Multiple Signal Drivers

- Remember: all concurrent statements are considered separate processes.
- Therefore, the following has multiple signal drivers:

Sally: process (A) begin
BUS <= not A;
end process;
---
BUS <= '1' when q=5 else '0';

Resolving Multiple Drivers

- What happens when two signal drivers drive the same wire to different states?
- Assume A, and BUS are:
  type MVL4 is ('X', '0', '1', 'Z');
  signal A, BUS: MVL4;
- This is simply an enumerated class
  • No inherent meaning in these symbols
- Solution: Must define a resolution function

Resolution Functions

- Signal Declaration Template
  signal sig_name: resolution_function type [:= initial_value];

- Resolution functions accept only one input parameter, which must be an unconstrained array type
An Example

- Wired 'X' Function- our arbitrary definition:
  - If one driver produces '1' and another driver produces '0', the result is an 'X'. Two 'Z's resolve to a 'Z'. '0' dominates 'Z'. '1' dominates 'Z'.

  type MVL4 is ('X', '0', '1', 'Z');
  type MVL4VECTOR is array (natural range <>) of MVL4;
  type MVL4_TABLE is array (MVL4,MVL4) of MVL4;

Example (cont'd)

- Define cross-reference table:
  - truth table for "wiredX" function
    constant TABLE_WIREDX: MVL4TABLE :=
    - | X 0 1 Z |
    - | X, X, X, X | - | X |
    - | X, 'X', 'X', 'X' | - | 0 |
    - | 'X', 'X, '1', '1', - | 1 |
    - | 'X, 'X, '1', 'Z' | - | Z |

- Resolution operation must be associative and commutative, i.e. order of evaluation is not important
- Pairwise resolution of signal values from multiple drivers

Example (cont'd)

- Define resolution function:
  function WIREDX (V: MVL4_VECTOR) return MVL4 is
    variable RESULT: MVL4 := 'Z';
    begin
      for i in V'range loop
        RESULT := TABLE_WIREDX(RESULT, V(i));
        exit when RESULT = 'X';
      end loop;
      return RESULT;
    end WIREDX;

How it works

- Signal driver sources
- WIRED_X
  - Resolved signal
  - Resolution function

Back to Example

architecture MULTIPLE of DRIVER is
signal BUS: WIREDX MVL4;
begin
Fred: process begin
  BUS <= '0', '1' after 10 ns, 'Z' after 20 ns;
  wait;
end process Fred;
Sam : process begin
  BUS <= '0', '1' after 5 ns, '0' after 15 ns;
  wait;
end process Sam;

Using Subtypes to Declare Resolved Signals

Subtype RESOLVED_MVL4 is WiredX MVL4;
signal BUS: RESOLVED_MVL4;