

IC/CAD Contest

Problem 5: Fully-Decoder Identification for the Bus-Contention Problem

Source: SynTest Technologies, Inc.

Feb. 15, 2000

I. Problem Description

In this problem, we are concerned about the fully-decoder identification for the bus-contention problem. Bus contention occurring in a design may damage the circuits. In the test phase, identifying all tri-state enable pins from the fully-decoder can reduce unnecessary cost in scan repairing. Engineers and testing tools pay much attention on this issue. A fully-decoder can guarantee that one and only one tri-state enable pin in a bus is active to avoid the bus contention problem. The objective of your tool is to determine whether each bus of a gate-level design is fully decoded.

II. Basic Specification

1. Input

Any gate-level design with buses using the Verilog language. To simplify the parser, the design can be a single module using the following primitives only: (a) **buf** and **not**, (b) 2-input **nand**, **nor**, **and**, **or**, **xor**, **xnor** and (c) **bufif0**, **bufif1**, **notif0** and **notif1**.

2. Output

Each bus in the design is fully decoded or not.

3. Examples

Three input examples (test cases) are given below. Your tool should be able to determine whether each of the buses in the test cases is fully decoded or not.

Test Case 1.

```
module sti_case1 ( X1, A1, A2, A3, A4, A5, A6, A7, A8 );
  output  X1;
  input   A1, A2, A3, A4, A5, A6, A7, A8;

  xor     ( c017, A5, A6 );
  xor     ( c018, A7, A8 );

  buf     ( c013, c017 );
  not     ( c014, c017 );
  buf     ( c015, c018 );
  not     ( c016, c018 );

  or      ( c006, c013, c015 );
  or      ( c008, c014, c015 );
  or      ( c010, c013, c016 );
  or      ( c012, c014, c016 );

  bufif0 ( c000, A1, c006 );
  bufif0 ( c000, A2, c008 );
```

```

bufif0 ( c000, A3, c010 );
bufif0 ( c000, A4, c012 );

buf    ( X1, c000 );
endmodule

```

Test Case 2.

```

module sti_case2 ( X1, A1, A2, A3, A4, A5, A6, A7, A8, S1, S2, S3, S4 );
output  X1;
input   A1, A2, A3, A4, A5, A6, A7, A8, S1, S2, S3, S4;

xor     ( c017, A5, A6 );
xor     ( c018, A7, A8 );

buf     ( c013, c017 );
not     ( c014, c017 );
buf     ( c015, c018 );
not     ( c016, c018 );

and     ( c006, c013, c015 );
and     ( c008, c014, c015 );
and     ( c010, c013, c016 );
and     ( c012, c014, c016 );

and     ( c005, S1, 1'b0 );
and     ( c007, S2, 1'b0 );
and     ( c009, S3, 1'b0 );
and     ( c011, S4, 1'b0 );

nor     ( c001, c005, c006 );
nor     ( c002, c007, c008 );
nor     ( c003, c009, c010 );
nor     ( c004, c011, c012 );

bufif0 ( c000, A1, c001 );
bufif0 ( c000, A2, c002 );
bufif0 ( c000, A3, c003 );
bufif0 ( c000, A4, c004 );

buf     ( X1, c000 );
endmodule

```

Test Case 3.

```

module sti_case3 ( X1, A1, A2, A3, A4, A5, A6, A7, A8, S1, S2, S3, S4 );
output  X1;
input   A1, A2, A3, A4, A5, A6, A7, A8, S1, S2, S3, S4;

xor     ( c017, A5, A6 );
xor     ( c018, A7, A8 );

buf     ( c013, c017 );
not     ( c014, c017 );

```

```

buf    ( c015, c018 );
not    ( c016, c018 );

and    ( c006, c013, c015 );
and    ( c008, c014, c015 );
and    ( c010, c013, c016 );
and    ( c012, c014, c016 );

and    ( c005, S1, 1'b1 );
and    ( c007, S2, 1'b1 );
and    ( c009, S3, 1'b1 );
and    ( c011, S4, 1'b1 );

nor    ( c001, c005, c006 );
nor    ( c002, c007, c008 );
nor    ( c003, c009, c010 );
nor    ( c004, c011, c012 );

bufif0 ( c000, A1, c001 );
bufif0 ( c000, A2, c002 );
bufif0 ( c000, A3, c003 );
bufif0 ( c000, A4, c004 );

buf    ( X1, c000 );
endmodule

```

III. Advanced Features

You are free to incorporate advanced features and functions in your tool. Some possible advanced features are as follows.

1. Besides the identification of bus contention, your tool may list the patterns that cause the bus contention.
2. You can extend your tool to parse more than 2-input primitives.
3. You can extend your tool to parse hierarchical modules.

IV. Language Platform

1. Language: C or C++.
2. Platform: SUN OS/Solaris or PC DOS/Windows.

V. Evaluation

Completeness, correctness, time complexity, memory requirement, advanced features/functions, user interface, etc.

VI. Questions

Please report any question regarding this problem to cad@cis.nctu.edu.tw with the email subject "CAD Contest: Problem 5." Your question(s) will be answered in two weeks, and the Q&A's will be posted at the contest web site

References

- [1] N. H. E. Weste and K. Eshraghian. *Principles of CMOS VLSI Design -- A Systems Perspective*. 2nd ed., Addison Wesley, NY, 1993.
- [2] M. Abramovici and M. A. Breuer and A. D. Friedman. *Digital Systems Testing and Testable Design*. IEEE Press, NY, 1990.
- [3] M. G. Arnold. *Verilog Digital Computer Design -- Algorithms into Hardware*. Prentice Hall, NJ, 1999.