```
// File: quick_unisim_test.v
// John Hubbard, 08 Mar 2015
//
// Using Modelsim to simulate your Xilinx designs can save accelerate your
// test-and-fix workflow by at least an order of magnitude, in many cases. That
// is because it takes seconds to rebuild and run in Modelsim, but minutes (or
// much more) to rebuild the FPGA bitstream from Vivado.
//
// However, if your design uses Xilinx primitives such as LUT4, LUT6, etc, then
// you'll need feed a Xilinx library (specifically, the "unisim" library) to
// Modelsim, in order for the simulation to work. Here are the steps to set
// that up, for Xilinx Vivado 2014.4, and Modelsim 2013.06.
//
// 1. Compile the simulation libraries, from within Vivado. This is done via:
//
//      Tools-->Compile Simulation Libraries
//
//     a) Choose "ModelSim" for the Simulator drop-down box (it should be there
//        already, as the default choice).
//
//     b) Be sure to specify an output directory, because otherwise, it can be a
//        little tricky to find out where Vivado put it. A reasonable choice for
//        the "Compiled library location" field is something like:
//
//            <your_verilog_projects_directory>/vivado_unisim_compiled
//
// 2. Within Modelsim: use the "import library" wizard to add the new library:
//
//      File-->Import-->Library
//
// , and browse to the following SUB-directory (easy to miss this point):
//
//      <your_verilog_projects_directory>/vivado_unisim_compiled/unisim_ver
//
// Modelsim's "import library" wizard will ask for a location to copy the
// imported library. A reasonable directory name for that is something like:
//
//      <your_verilog_projects_directory>/vivado_unisim_imported
//
// 3. When linking (running vsim), pass in the library as follows:
//
//     vsim work.quick_test -L unisims_ver
//
// References:
// [1] Vivado Simulation Guide: ug900-vivado-logic-simulation.pdf:
// http://www.xilinx.com/support/documentation/sw_manuals/xilinx2014_4/ug900-vivado-logic-
simulation.pdf
//----------------------------------------------------------------------------
// EXAMPLE:
//
//  vlog ../quick_unisim_test.v
//  # Model Technology ModelSim ALTERA vlog 10.1e Compiler 2013.06 Jun 12 2013
//  # -- Compiling module quick_unisim
//  # -- Compiling module quick_test
```

```
//  #
//  # Top level modules:
//  #   quick_test
//  vsim work.quick_test -L unisims_ver
//  # vsim -L unisims_ver work.quick_test
//  # Loading work.quick_test
//  # Loading work.quick_unisim
//  # Loading unisims_ver.LUT2
//  # Loading unisims_ver.x_lut2_mux4
//  run 10 ns
//  # a, b, f: 0, 0 | 0
//  # a, b, f: 0, 1 | 0
//  # a, b, f: 1, 0 | 0
//  # a, b, f: 1, 1 | 1
//------------------------------------------------------------------------------

`timescale 1ns/1ns

module tiny_unisim_example(a, b, f);
    input a, b;
    output f;

    // A LUT mask of 0x8 creates an AND gate:
    LUT2 #(16'h8) theLUT(f, a, b);
endmodule

module quick_test();
    reg a, b;
    reg [4:0] vec;
    wire f;
    parameter max_vec = 4;

    tiny_unisim_example DUT(a, b, f);

    initial
    begin
        for( vec = 0; vec < max_vec; vec = vec +  1 )
        begin
            {a,b} = vec;
            #1;
            $display("a, b, f: %b, %b | %b", a, b, f);
        end
    end
endmodule
```