COMPUTER SCIENCE TRIPOS Part IB – 2014 – Paper 5

1 Computer Design (SWM)

A novice SystemVerilog programmer has written the following decimal counter module which should zero the `decimal_count` on reset and then, when enabled, increment modulo 10 the `decimal_count` on every positive clock edge.

```
module count_decimal_wrong(
    input  logic clk;
    input  logic reset;
    input  logic enable;
    output logic decimal_count);

   always_comb @(posedge clk or reset)
     if(enable)
       begin
          decimal_count = decimal_count+1;
          if(decimal_count>9)
            decimal_count = 0;
       end
     elsif(reset)
       decimal_count = 0;

endmodule // count_decimal_wrong
```

(*a*) What bugs exist in the code and how can they be rectified? [10 marks]

(*b*) SystemVerilog synthesis tools use a Boolean optimiser to simplify the implementation logic.

(*i*) Why are *don't care* terms useful for Boolean optimisation? [3 marks]

(*ii*) How could the SystemVerilog be modified to introduce *don't care* terms for unreachable states above 9? [3 marks]

(*iii*) For a modern FPGA with 6-input look-up tables (LUTs), will Boolean optimisation result in fewer resources being used for the corrected `count_decimal`? Justify your answer. [4 marks]