Coláiste na Tríonóide, Baile Átha Cliath
Trinity College Dublin
Ollscoil Átha Cliath | The University of Dublin

FACULTY OF ENGINEERING, MATHEMATICS & SCIENCE

SCHOOL OF ENGINEERING

Electronic & Electrical Engineering

Engineering — Semester 2

Junior Sophister

Annual Examinations

Digital Systems Design (EE3C07)

23rd April 2019 — RDS SIMMONSCOURT — 14.00 – 16.00

Associate Professor Naomi Harte

Instructions to Candidates:

Answer FOUR questions in total. Question 1 is COMPULSORY.

Answer any THREE of the remaining questions.

Materials Permitted for this Examination:

Calculator

Drawing Instruments

Graph Paper

Formulae and Tables are available from the invigilators, if required

Non-programmable calculators are permitted for this examination. Please indicate the make and model of your calculator on each answer book used..

Q.1

a) Why do Moore Finite State Machines typically have more states that an equivalent Mealy Machine?

b) What is the potential consequence of not observing the set-up time of a register in an FPGA design?

c) Why is the use of "connection by name" instantiation a useful coding practice in Verilog?

d) A Verilog module contains the following snippet of code:

```
always @(posedge clk, posedge reset)
   if (reset)
      q <= 0;
   else
      q <= d;
```

What is the proper name given to this reset strategy and how will it behave in reset?

e) An engineer is targeting a new product at a high-volume time-critical market. Should she prototype on ASIC or FPGA? Briefly explain.

f) Give 3 desirable properties of a testbench in Verilog.

g) Why is the single-cycle multiply-accumulate (or MAC) capability of DSP processors so important in DSP filters?

h) Write down a forbidden reset value for a 7-bit LFSR using XNOR feedback.

i) Is the silicon layout stage of a typical ASIC design flow necessary in FPGA design? Briefly explain.

j) Explain how to detect overflow in the output of a 4-bit carry-ripple adder

[25 marks, equally divided]

Q.2

A data transmission system has two input channels, A and B, each receiving synchronous serial data. You are required to design a single finite state machine (FSM) to flag whenever the codeword "1101" is detected in both channels simultaneously.

(a) Design a FSM to control the machine. Clearly explain how the FSM operates, supported by a clearly labelled state diagram. Is it a Moore or a Mealy state machine? Complete a state table for your state machine.

[8 marks]

(b) Using an implication chart to examine the state table, systematically ensure your FSM has the minimum number of states required. Draw the final state machine diagram, clearly indicating inputs, outputs and transitions.

[8 marks]

(c) Presume that your design is implemented in a Verilog module (no code required). Presume that the internal state of the FSM is registered and that all outputs of the module are also registered. Sketch a timing diagram that demonstrates the behaviour of your state machine and associated outputs upon encountering the sequence 001110100 on both channels. Assume a single global clock signal.

[9 marks]

Q.3

Consider the Verilog module below and answer the questions that follow overleaf:

```
module sample_lfsr                                                    1
(input wire clk, reset,                                               2
output wire lfsr_out_reg, max_tick_reg);                              3
                                                                      4
reg [8:0] lfsr_reg;                                                   5
reg [8:0] lfsr_next;                                                  6
reg lfsr_tap;                                                         7
reg lfsr_out_next;                                                    8
reg max_tick_next;                                                    9
                                                                      10
always @(posedge clk, posedge reset)                                  11
  if (reset)                                                          12
      begin                                                           13
          lfsr_reg <= 9'b11100_0011;                                  14
          lfsr_out_reg <= 1'b1;                                       15
          max_tick_reg <= 1'b0;                                       16
      end                                                             17
  else                                                                18
      begin                                                           19
          lfsr_reg <= lfsr_next;                                      20
          lfsr_out_reg <= lfsr_out_next;                              21
          max_tick_reg <= max_tick_next;                              22
      end                                                             23
                                                                      24
always @*                                                             25
  begin                                                               26
      lfsr_tap = lfsr_reg[8] ^ lfsr_reg[4];                           27
      lfsr_next = {lfsr_reg[7:0],lfsr_tap };                          28
  end                                                                 29
                                                                      30
assign lfsr_out_next = lfsr_reg[8];                                   31
assign max_tick_next = (lfsr_reg == 9'b11100_0011);                   32
                                                                      33
endmodule                                                             34
```

Q.3 (continued from previous page)

(a) Sketch a block diagram that shows how the LFSR, described in the Verilog code provided, operates. Clearly identify the next-state logic and registers in the design.

[8 marks]

(b) Rewrite line 31 and 32 using procedural statements. If you made this change to the module, what other code changes would be required?

[6 marks]

(c) With direct reference to the operation of the clocked always block in the supplied code, compare the use of blocking versus non-blocking statements in Verilog.

[6 marks]

(d) Give 3 practical applications for LFSRs. Why advantages do they have over standard binary counters?

[5 marks]

Q.4

(a) Consider the Master-Slave D-type flip flop in Figure Q4a. Outline the operation of the latch in detail, and in the process explain whether the device is overall edge sensitive or level sensitive.

[12 marks]

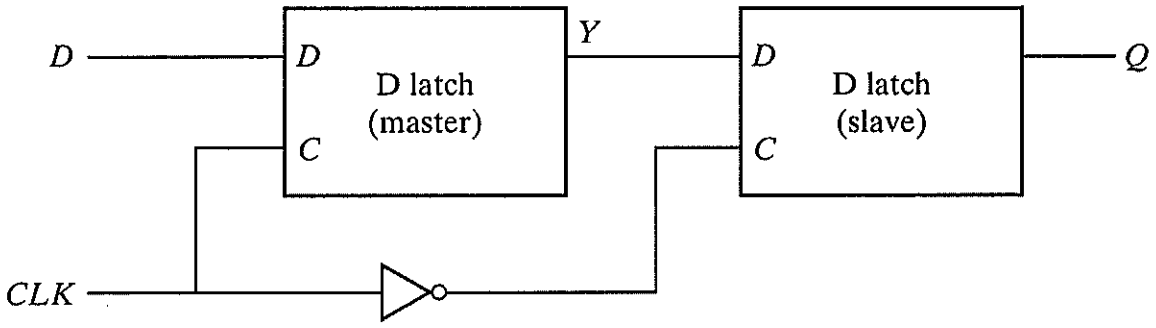

Figure Q4a

(b) Consider the input D as shown in Figure Q4b on the next page. Complete the timing diagram, clearly showing the outputs Y and Q. Also indicate when the master and slave latches are active.

[13 marks]

[note Figure Q4b on next page.....]

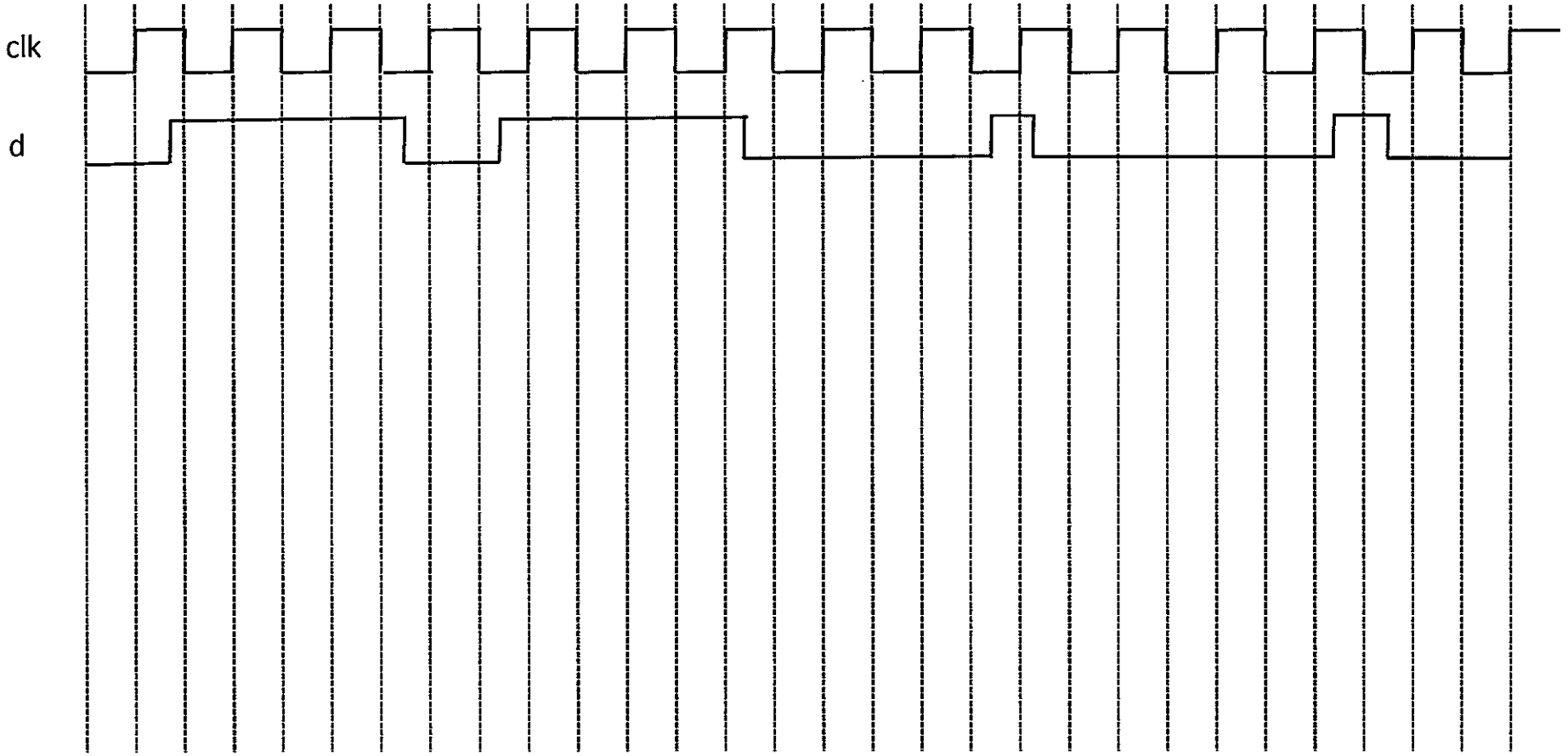

Figure Q4b (Question 4 is on previous page)

Include Exam ID if submitting this sheet with your script: _____

Q.5

(a) As part of the 3C7 course, you designed an ALU. There was a requirement to only have instantiations in the top level module, with no additional logic. Explain why this coding requirement is good practice.

[8 marks]

(b) Outline the design flow you typically encounter in 3C7, taking a design from concept through to implementation on the FPGA board. Support your answer with a diagram.

[7 marks]

(c) Explain the purpose of the synthesis step in FPGA development. How does simulation after synthesis differ from pre-synthesis simulation?

[5 marks]

(d) Compare and contrast place & route in a typical ASIC versus FPGA design flow.

[5 marks]