UNIVERSITY OF RUHUNA

Faculty of Engineering

End-Semester 8 Examination in Engineering: November 2016

Module Number: EE8214

Module Name: Introduction to Hardware Description Languages

[Three Hours]

[Answer **all questions**, each question carries 10 marks]

Q1 a) i) Briefly Explain the importance of using Hardware Description Languages (HDLs).

ii) What is the difference between the two data types 'wire' and 'reg' from synthesis perspective?

iii) What is the difference between the blocking and non-blocking statements?

[3 Marks]

b) Write Verilog code for a Full Adder using following modeling styles.

i) Gate level modelling

ii) Behavioral modelling

iii) Structural modelling - use half adders

[3 Marks]

c) Clearly mention whether the following constructs are synthesizable or not. Give your reasons.

i)
```
wire a,b;
assign a = b;
```

ii)
```
initial begin
clk = 0;
end
```

iii)
```
wire q, x;
always @(negedge clk)
q = x;
```

iv)
```
$monitor ("%g a = %b", $time, a);
#10 a =0;
#11 a = 1;
#12 a = 0;
#13 a = 1;
#14 $finish;
```

[4 Marks]

Q2 a) i) Explain the difference between synchronous sequential circuits and asynchronous sequential circuits.

ii) Write Verilog Register Transfer Level (RTL) and explain synchronous reset and asynchronous reset.

[4 Marks]

b) Answer the following questions based on the given fragment of Verilog HDL.

i) Determine when the variable 'c' is changing and the changed value of it referring the below code fragment.

```
initial
begin
a <= 0;
b<=1;
#20 a <= 1;
b <= #20 a + 1;
c <= #20 b + 1;
end
```

ii) Describe the following Verilog code fragment.

```
reg [7:0] my_signal;
  always @ (posedge clock) begin
   my_signal <= my_signal << 1;
end
```

iii) Describe the following Verilog code fragment.

```
always @(posedge CLK or posedge CLR)
begin
    if (CLR)
        Q <= 1'b0;
    else
        Q <= IN;
end
```

[6 Marks]

Q3 Nowadays, designers use multiple clocks having different frequencies in System on Chip (SoC) designs due to many reasons. This usage of multiple clocks creates multiple clock domains in the design. When signals transfer through different domains, lot of issues might raise in a design. These issues are called "Clock Domain Crossing (CDC) issues". Therefore, CDC verification has become one of the major verification challenges in SoC design flow.

a) Metastability is one of the major CDC issues.

i) What is CDC?

ii) Explain why Metastabilty happens when there is a CDC.

iii) Draw a diagram to illustrate how Metastability happens at a CDC.

iv) State two harmful impacts that can occur in a design due to Metastability.

[6 Marks]

b) To overcome Metastability, designers use different synchronization schemes.

i) State five synchronization schemes.

ii) Draw diagrams of two synchronization schemes that you have mentioned above.

[4 Marks]

Q4 a) i) What are the advantages of Field Programmable Gate Array (FPGA) compared to Application Specific Integrated Circuit (ASIC)?

ii) Explain FPGA-based prototyping concept and its importance.

iii) What is the difference between code coverage and functional coverage? [Hint: Consider the purpose and support languages etc. for your answer]

iv) Figure Q4 a) shows how the Configurable Logic Blocks (CLB) are connected with programmable interconnects inside a typical FPGA. Draw the diagram of CLB showing the combinational and sequential blocks.

[6 Marks]

b) Consider the following RTL code fragment.

```
module test(clk,in1,in2,out1,out2);
input clk ;
input in1;
input [3:0] in2;
output out1;
output [3:0] out2;
reg [3:0] out2;
assign out1 = (in1) ? 1'b0:1'b1 ;
always @(posedge clk) begin
case(in2)
4'b0000: out2 <= 1;
4'b0001: out2 <= 2;
4'b0010: out2 <= 3;
default: out2<= 4;
endcase
end
endmodule
```

i) What are the statements for the line coverage?

ii) What are the signals considered for the toggle coverage?

iii) What are the considered stares for the Finite State Machine (FSM) coverage?

iv) State the conditional coverage conditions.

[4 Marks]

Q5 Synthesis is the process of transforming the behavior and the structure of electronic circuits described using high level HDL constructs into low level logical constructs called standard cells that can be literally modeled in the form of actual physical hardware.

a) Draw the schematic diagram for the following RTL modules using standard schematic symbols.

i)
```
module top (A, B, X, Z);
input A, B;
output X, Z;
assign X = A | B;
assign Z = A & B;
endmodule
```

ii)
```
module top (clk, A, B, X, Z);
input clk, A, B;
output reg X, Z;
always @ (posedge clk) begin
Z = A | B;
X = A  & B;
end
endmodule
```

iii)
```
module top (sel, A, B, Z);
input sel, A, B;
output reg Z;
always @ (sel or A or B) begin
if (sel == 1'b0)
                    Z = A | B;
else
                    Z = A & B;
end
endmodule
```

iv)
```
module top (clk, sel, A, B, X, Z);
input clk, sel, A, B;
output reg X, Z;
wire W1;

assign W1 = sel ? A | B  : A & B;
always @ (posedge clk) begin
X <= W1 | B;
Z <= X & A;
end
endmodule
```

[8 Marks]

b) Draw the corresponding FSM defined in following Verilog RTL.
[Hint: You can neglect the reset function defined in RTL]

```
module top (clk, reset, data_in, out);
input clk, reset, data_in;
output reg out;
reg [2:0]present_state,next_state;
```

```
parameter s0=0, s1=1;

always @(posedge clk)begin
  present_state = next_state;
end
always @(present_state or data_in or reset)begin
  if(reset)begin
    next_state = s0;
    out = 0;
  end
  else begin
    case(present_state)
    s0:begin
      if(data_in == 1'b0)begin
      next_state = s1;
      out = 0;
      end
      else begin
      next_state = s0;
      out = 0;
      end
    end
    s1:begin
      if(data_in == 1'b0)begin
      next_state = s1;
      out = 0;
      end
      else begin
      next_state = s0;
      out = 1;
      end
    end
    endcase
  end
end
endmodule
```

[2 Marks]

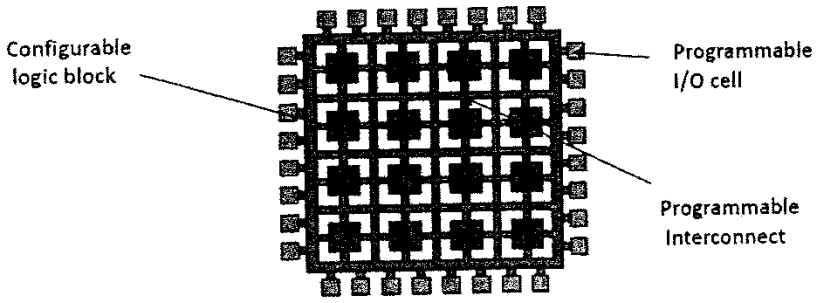

Figure Q4 a)