Week 10: Assignment 10

The due date for submitting this assignment has passed.

Due on 2021-10-06, 23:59 IST.

As per our records you have not submitted this assignment.

1) Assume R_c represents the condition of execution and r_c represents the data transformation for a computation c in a finite state machines with datapaths (FSMD). Then, two computations c1 and c2 are equivalent if *1 point*

- $R_{c1} = R_{c2}$ and $r_{c1} != r_{c2}$
- $R_{c1} != R_{c2}$ and $r_{c1} = r_{c2}$
- $R_{c1} = R_{c2}$ and $r_{c1} = r_{c2}$
- $R_{c1} != R_{c2}$ and $r_{c1} != r_{c2}$

No, the answer is incorrect.
Score: 0
Accepted Answers:
$R_{c1} = R_{c2}$ and $r_{c1} = r_{c2}$

2) What would be the total number of computations for the FSMD generated from the code snippet given below? *1 point*

```
if(c1){
    if(c2)
        t1 = a + b;
    else
        t1 = c x d;
}
else{
    t1 = c x d;
}
```

- 1
- 2
- 3
- 4

No, the answer is incorrect.
Score: 0
Accepted Answers:
3

3) What would be the total number of paths and computations for the FSMD M1, respectively? Consider q0 as the reset state or start state and nodes with more than one outgoing edge as cut-points. (Note: A computation is an execution of the behaviour and a path is between two cut-points without having any intermediate cut-point) *2 points*

- 3, 2
- 6, 2
- 3, 8
- 6, 8

No, the answer is incorrect.
Score: 0
Accepted Answers:
6, 8

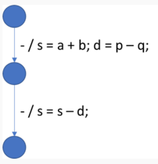
4) What would be the correct conversion of c code from the Verilog code given below. Consider a, c, x, y and m are registers. *2 points*

```
always @ (posedge ap_clk) begin
    if (1'b1 == ap_CS_fsm_state2) begin
    a <= b;
    c <= d;
    end
end
assign b = x + y;
assign d = a + m;
```

- ap_CS_fsm_state2;
  x_old = x;
  y_old = y;
  if (1 == ap_CS_fsm_state2)
  {
  a = x + y;
  }
  if (1 == ap_CS_fsm_state2)
  {
  c = a + m;
  }
- ap_CS_fsm_state2;
  b_old = b;
  d_old = d;
  if (1 == ap_CS_fsm_state2)
  {
  a = b_old;
  c = d_old;
  }
  if (1 == ap_CS_fsm_state2)
  {
  a = x + y;
  c = a + m;
  }
- ap_CS_fsm_state2;
  a_old = a;
  x_old = x;
  y_old = y;
  m_old = m;
  c_old = c;
  if (1 == ap_CS_fsm_state2)
  {
  a = x_old + y_old;
  }
  if (1 == ap_CS_fsm_state2)
  {
  c = a_old + m_old;
  }
- None of the above

No, the answer is incorrect.
Score: 0
Accepted Answers:
ap_CS_fsm_state2;
a_old = a;
x_old = x;
y_old = y;
m_old = m;
c_old = c;
if (1 == ap_CS_fsm_state2)
{
a = x_old + y_old;
}
if (1 == ap_CS_fsm_state2)
{
c = a_old + m_old;
}

5) Consider the following path. What would be the final symbolic value of s at the end of the path? *1 point*



- a + b
- s - d
- a + b - d
- a + b - p + q

No, the answer is incorrect.
Score: 0
Accepted Answers:
a + b - p + q

6) Consider the following two statements. Which one is TRUE about verification of High-level Synthesis? *1 point*

S1: Formal verification checks the equivalence of inputs and outputs of HLS symbolically; it does not need any test case.

S2: Simulation based verification of guarantees 100% correctness of the High-level Synthesis.

- Only S1 is True
- Only S2 is True
- Both S1 and S2 are True
- Both S1 and S2 are False

No, the answer is incorrect.
Score: 0
Accepted Answers:
Only S1 is True

7) Reverse engineering an equivalent C code from the HLS generated RTL helps in faster simulation based verification of HLS. *1 point*

- True
- False

No, the answer is incorrect.
Score: 0
Accepted Answers:
True

8) Which one of the following statements is correct? *1 point*

S1. RTL simulation is faster than C simulation.

S2. Phase wise verification of HLS needs intermediate information from the HLS tool.

S3. It is not possible to extract a C like behaviour from the RTL generated by the HLS tool.

- Only S1
- Only S2
- Only S3
- None of them

No, the answer is incorrect.
Score: 0
Accepted Answers:
Only S2