Session 3 Exercise

fares.sultan9@gmail.com Switch account

Not shared

* Indicates required question

Name *

Choose

What is functional coverage in SystemVerilog? * 1 point

- Measurement of timing delays
- Measurement of code execution paths
- Measurement of the design intereseting transitions/values has been exercised
- Measurement of correctness of the DUT

You are required to constraint an active low signal to be deactivated most of simulation time which one of the choices represents that case? * 1 point

- 0 := 70 , 1 := 30
- 0 := 30 , 1 := 70

What is the significance of a bins construct in SystemVerilog functional coverage? * 1 point

- To count the number of errors
- To represent possible values to monitor for coverpoints
- To define the number of test cases
- To simulate clock signals

Which of the following can be used to if we have invalid values in coverage? * 1 point

- exclude
- ignore
- coverpoint none
- illegal_bins

How do you access the last element of a queue in SystemVerilog? * 1 point

- queue[0]
- queue.first()
- queue[$]
- queue[-1]

Which of the following methods is used to insert an element at the front of a queue? * 1 point

- push_front()
- push_back()
- insert()
- enqueue()

How do you access the number of elements in a queue? * 1 point

- queue.count()
- queue.length()
- queue.size()
- none

Which method removes the last element of a queue in SystemVerilog? * 1 point

- pop_back()
- pop_last()
- dequeue()
- push_back()

If you reached 100% functional coverage, then you do not need to check the code coverage. * 1 point

- ◯ True
- ◯ False

If u need to write a coverpoint for some values (ex : 0 , 1 , 3) for variable X, is this correct? * 1 point

```
operand_1:coverpoint X{
bins test_bin=(0,1,3);}
```

- ◯ True
- ◯ False

If u need to do a transaction between ADD to SUB : * 1 point

```
bins ADDSUB={ADD => SUB};
```

- ◯ Yes
- ◯ No

complete the following code : * 1 point

```
constraint test {
foreach( .....(1)..... )
    .....(2)..... dist {0 :/80 , 1 :/ 20 };
    }
```

- (1) array [i][j] , (2) array [i][j]
- (1) array [i , j] , (2) array [i][j]
- (1) array [i][j] , (2) array [i , j]
- (1) array [i , j] , (2) array [i , j]

Covergroup can be in ..... * 1 point

- a class inside package only
- a module only
- Both

in the following code it mean * 1 point

```
bins test_tr = (5 => 1 , 4);
```

- transaction from 5 to 1 to 4
- transaction from 5 to 1 or transaction from 5 to 4
- values between 5 and 1 exclude 4

if (red_op_A && red_op_B) both high , which constrain will occur * 1 point

```
[1] if ((opcode == OR || opcode == XOR) && red_op_A) {
        A dist {{3'b001, 3'b010, 3'b100}:/75, {3'b011, 3'b101, 3'b110, 3'b111}:/15};
        B == 3'b000;
    }

[2] if ((opcode == OR || opcode == XOR) && red_op_B) {
        B dist {{3'b001, 3'b010, 3'b100}:/75, {3'b011, 3'b101, 3'b110, 3'b111}:/15};
        A == 3'b000;
    }
```

- ◯ [1]
- ◯ [2]
- ◯ Conflict
- ◯ None

You should take care of the constraints order because they aren't resolved in parallel * 1 point

- ◯ True
- ◯ False

In-line constraints have higher priority on the class constraint blocks * 1 point

- ◯ True
- ◯ False
- ◯ True if you defined the constrained variable using soft keyword in the constraint block

You can always loop on an array in a constraint block * 1 point

- True
- True but only using foreach
- None of the above

Post and pre-randomize functions are * 1 point

- called when creating a new object
- called when using the .randomize built-in method
- All of the above

You can only turn on or off all the class properties (variables) randomization but not some of them * 1 point

- True
- False

Covergroups can contain more than cover point for more than one variable * 1 point

- True
- False

X: coverpoint Y {....} *
The previous code line can be interpreted as: 1 point

- X: is comment and Y is the coverpoint name
- X is the coverpoint label and Y is the covered variable
- X is the covergroupname and Y is the coverpoint name

The size of associative arrays is determined * 1 point

- During the array declaration
- When assigning a value to a new address the array size increases
- Through arrayname.push method

The implication operator used in the constraint block is similar to using ... in the constraint block * 1 point

- weighted distribution
- inline constraint
- if

What does the inline constraints provide the testbench creator? * 1 point

- The ability to turn off constraints
- The ability to provide more constraints with the constraints defined the constraint blocks of the class
- The ability to turn off randomization

Which method is considered a must to constraint for the dynamic arrays? * 1 point

- .size
- .sum

Which is the easiest method to create a scenario generated stimulus in a class-based testbench? * 1 point

- Using arrays
- Using .randomize
- Using pre/post-randomize
- Using Constraint blocks

rand_mode can .. * 1 point

- enable/disable randomization for all class variables
- enable/disable randomization for certain class variables
- Both

constraint_mode can .. * 1 point

- enable/disable all class constraint blocks
- enable/disable certain class constraint blocks
- Both

The available bins to use in a coverpoints are * 1 point

- [ ] legal bins
- [ ] ignore bins
- [ ] exclude bins
- [ ] illegal bins

Which syntax correctly applies an inline constraint? * 1 point

- ( ) x.randomize { x inside {[1:10]}; };
- ( ) constraint { x inside {[1:10]}; };
- ( ) randvar x inside {[1:10]};
- ( ) randomize(x) with { x inside {[1:10]}; };

What does randmode(0) do in SystemVerilog? * 1 point

- ( ) Creates a new randomization mode
- ( ) Disables randomization for a variable
- ( ) Enables randomization for a variable
- ( ) Deletes a constraint

When would you use constraint_mode(1)? * 1 point

- To delete all constraints
- To check for conflicting constraints
- To remove a variable from randomization
- To re-enable a previously disabled constraint

Which method is executed before the randomize() method? * 1 point

- randomize()
- pre_randomize()
- randcase()
- post_randomize()

Which of the following is a valid functional coverage construct? * 1 point

- covergroup
- cross
- coverpoint
- All of the above

What is a legal bin in functional coverage? * 1 point

- A bin that does not contribute to coverage
- A bin that collects valid scenarios
- A bin that is ignored during simulation
- A bin that stores only illegal cases

What is the purpose of ignore bins in functional coverage? * 1 point

- To exclude specific values from coverage
- To mark values as illegal
- To generate new random values
- To disable functional coverage

What is the purpose of illegal bins in functional coverage? * 1 point

- To enable cross-coverage
- To increase coverage
- To mark invalid values for coverage analysis
- To remove constraints

What does cross coverage in SystemVerilog measure? * 1 point

- The probability of constraints being met
- The occurrence of illegal bins
- The ratio of randomization failures
- The interaction between multiple coverpoints

What type of data structure is an associative array in SystemVerilog? * 1 point

- A dynamically allocated linked list
- A fixed-size array
- A queue-like structure
- An array indexed by keys instead of numeric indices

What is the purpose of the implication operator (->) in constrained randomization? * 1 point

- To define a range of values
- To specify a condition that must be true for another constraint to apply
- To generate random values without constraints
- To define illegal values

In the constraint if (a == 1) -> b == 0, what happens if a is not equal to 1? * 1 point

- ○ The constraint is ignored
- ○ b must still be 0
- ○ An error is generated
- ○ a is forced to be 1

What is the scope of inline constraints? * 1 point

- ○ They apply globally to all randomizations
- ○ They apply only to the current randomization call
- ○ They override all class constraints
- ○ They are permanent and cannot be changed

What is an associative array in SystemVerilog? * 1 point

- ○ An array with a fixed size and indexed by integers.
- ○ An array that uses a key-value pair for indexing.
- ○ An array that can only store strings.
- ○ An array that is automatically resized when elements are added.

What will be the output of the following code? * 1 point

```
int my_array[int];
my_array[5] = 50;
my_array[10] = 100;
$display("%0d", my_array.num());
```

- 0
- 1
- 2
- 10

Which of the following is NOT a valid operation on an associative array? * 1 point

- Iterating through all elements using a foreach loop.
- Sorting the array in ascending order.
- Accessing elements using a string key.
- Resizing the array dynamically.

Which of the following is true about the memory usage of associative arrays? * 1 point

- They consume memory only for the elements that are explicitly assigned.
- They consume memory for all possible key values.
- They consume a fixed amount of memory regardless of the number of elements.
- They cannot be used in simulation due to high memory usage.

What will be the output of the following code? * 1 point

```
int my_array[int];
my_array[5] = 50;
my_array[10] = 100;
int key = 5;
key = my_array.next(key);
$display("%0d", key);
```

- 5
- 10
- 50
- 100

What will be the output of the following code? initial begin * 1 point

```
 fork
 #10 $display("Thread 1");
 #20 $display("Thread 2");
 #30 $display("Thread 3");
 join_none
 #5 $display("Main Thread");
end
```

- Main Thread Thread 1 Thread 2 Thread 3
- Thread 1 Main Thread Thread 2 Thread 3
- Main Thread
- Thread 1 Thread 2 Thread 3 Main Thread

What will be the output of the following code? * 1 point

```
initial begin
 fork
 begin
 #10 $display("Thread 1");
 #10 $display("Thread 2");
 end
 #15 $display("Thread 3");
 join
 $display("Main Thread");
end
```

- Thread 1 Thread 2 Thread 3 Main Thread
- Thread 1 Thread 3 Thread 2 Main Thread
- Thread 1 Thread 2 Main Thread Thread 3
- Thread 3 Thread 1 Thread 2 Main Thread

What will happen if two variables in a cross coverage have 4 and 3 bins, respectively? * 1 point

- The cross coverage will have 7 bins.
- The cross coverage will have 12 bins.
- The cross coverage will have 1 bin.
- The cross coverage will have 4 bins.

What will be the output of the following code? * 1 point

```
bit [1:0] a;
bit [1:0] b;
covergroup cg;

 coverpoint a { bins a_bins[] = {[0:3]}; }
 coverpoint b { bins b_bins[] = {[0:3]}; }
 cross a, b { illegal_bins illegal = binsof(a) intersect {2} && binsof(b)
intersect {2}; }

endgroup
cg cg_inst = new();
```

- 4 bins for a, 4 bins for b, and 16 cross coverage bins.
- 4 bins for a, 4 bins for b, and 15 cross coverage bins.
- 4 bins for a, 4 bins for b, and 12 cross coverage bins.
- 4 bins for a, 4 bins for b, and 1 cross coverage bin.

What will be the result of the following cross coverage definition? * 1 point

```
bit [1:0] a;
bit [1:0] b;
covergroup cg;

 coverpoint a { bins a_bins[] = {[0:1]}; }
 coverpoint b { bins b_bins[] = {[0:1]}; }
 cross a, b { ignore_bins ignore = binsof(a) intersect {2}; }

endgroup
cg cg_inst = new();
```

- ◯ 2 bins for a, 2 bins for b, and 4 cross coverage bins.
- ◯ 2 bins for a, 2 bins for b, and 3 cross coverage bins.
- ◯ 2 bins for a, 2 bins for b, and 2 cross coverage bins.
- ◯ 2 bins for a, 2 bins for b, and 1 cross coverage bin.

Submit Page 1 of 1 Clear form