Module: SV for Verification
Section: Randomization **Task:** Constraints

Task 1

Constraints

➢ **How you can turn on and off constraints in SystemVerilog?**
In SystemVerilog, you can control whether a constraint is applied during randomization using the **constraint_mode()** method. This method allows you to enable or disable specific constraints.

When a constraint is disabled (inactive), it will be ignored by the **randomize()** method, meaning it won't influence the randomization process. By default, all constraints are active, but you can selectively turn them off when needed to achieve different test scenarios or to focus on specific aspects of your design.

➢ **What is meant by weighted distribution in constraint randomization and what is the operator that is used for it?**
In SystemVerilog constraint randomization, **weighted distribution** refers to the ability to influence the likelihood that certain values are chosen during the randomization process. By assigning different weights to different possible outcomes, you can control how frequently those outcomes are selected.

For example, if you want one value to be chosen more often than another, you can assign a higher weight to that value. This allows for more nuanced testing by focusing on scenarios that might be more critical or by ensuring that edge cases are still covered but with a lower probability.

The operator used for weighted distribution is the **"dist"** operator. This operator allows you to specify different weights for different values or ranges of values.

- **Example:**

```
rand bit [3:0] priority;

constraint priority_c {
    priority dist {4'd0 := 10, 4'd1 := 30, 4'd2 := 20, [4'd3:4'd5] := 5, 4'd6 :=
25};
}
```

In this example, the variable priority can take on values between 0 and 6, but each value is weighted differently. For instance, **'4'd1'** is more likely to be chosen because it has a weight of 30, while values in the range **[4'd3:4'd5]** are less likely because they are given a weight of 5.

➢ **How can you randomize an array? Elaborate your answer with an example.**
In SystemVerilog, you can randomize an array by using the **randomize()** method, just as you would with scalar variables. To randomize an array, you typically define it as a **rand array** within a class, and then apply constraints to it if necessary. The randomize() method will then generate random values for the array elements, considering any constraints you've specified.

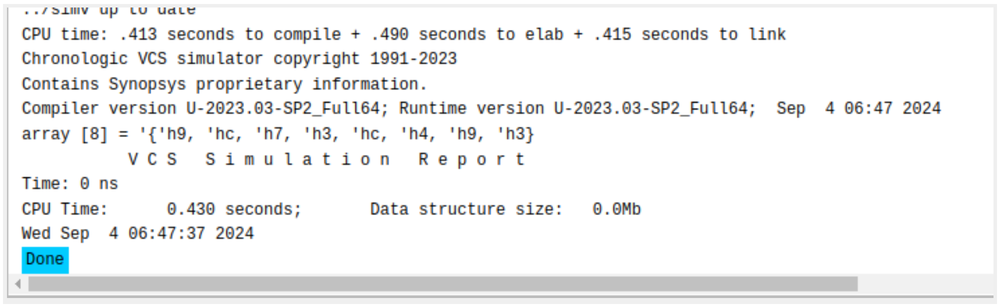
- **Example Code Snippet:**

```
class Packet;
  rand bit [3:0] array[8];
endclass

module test;
  Packet pkt;

  initial begin
    pkt = new();
    pkt.randomize();
    $display("array [8] = %p", pkt.array);
  end
endmodule
```

- **Output:**



```
../simv up to date
CPU time: .413 seconds to compile + .490 seconds to elab + .415 seconds to link
Chronologic VCS simulator copyright 1991-2023
Contains Synopsys proprietary information.
Compiler version U-2023.03-SP2_Full64; Runtime version U-2023.03-SP2_Full64;  Sep  4 06:47 2024
array [8] = '{'h9, 'hc, 'h7, 'h3, 'hc, 'h4, 'h9, 'h3}
           V C S   S i m u l a t i o n   R e p o r t
Time: 0 ns
CPU Time:      0.430 seconds;       Data structure size:   0.0Mb
Wed Sep  4 06:47:37 2024
Done
```