**Module: SV for Verification**

**Section:** Threads & Interprocess Communication **Task:** Fork Disable

**Task**

Fork Disable

* **Code:**

module fork\_disable;

initial begin

fork

begin // process A

$display("Process A started at time = %0t", $time);

#10;

$display("Process A completed at time = %0t", $time);

end

begin // process B

$display("Process B started at time = %0t", $time);

#15;

$display("Process B completed at time = %0t", $time);

end

begin // process C

$display("Process C started at time = %0t", $time);

#20;

$display("Process C completed at time = %0t", $time);

end

join\_any

//Add code here for disabling fork

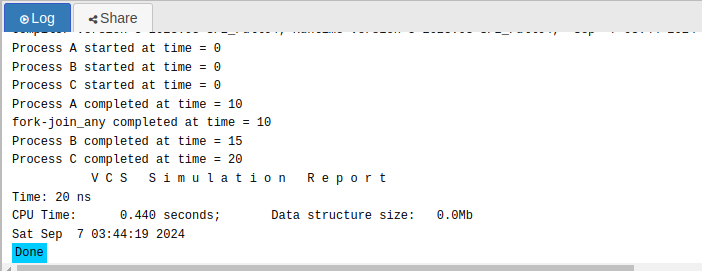
disable fork;

$display("fork-join\_any completed at time = %0t", $time);

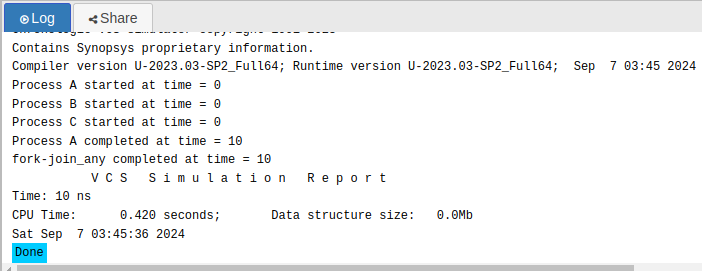
end

endmodule

* **Before Fork Disable:**
  + **Output:**



* **After Fork Disable:**
  + **Output:**



* **Observation:**

In this code, using **join\_any** causes the program to proceed as soon as any one of the forked processes (A, B, or C) completes. The first process to finish, which in this case is **Process A** after 10 time units, allows the program to move on immediately. The **disable fork;** statement ensures that after one process completes, the other forked processes are terminated, preventing them from finishing their execution as they did when there was no **disable fork;** statement in our code.

As a result, **Process B** and **Process C** are never fully executed. This leads to the premature ending of the remaining tasks, and the message "fork-join\_any completed" is printed at 10 time units when **Process A** completes.