SystemVerilog Assignment

Name: Shefin Muhammed MS

Systemverilog Lab2 : Procedural Statements & Routines

1. Procedural Statements & Routines

1.1. New procedural statements and operators

Code:

```
module example_loop;
  initial
  begin : example
    integer array[10], sum, j;

    // Fill array with values 0 to 9
    for (int i = 0; i < 10; i++)
      array[i] = i;

    // Start summing from last element
    sum = array[9];
    j = 8;

    // do...while loop to accumulate remaining elements
    do begin
      sum += array[j];
    end while (j--);

    // Display final result
    $display("Sum = %4d", sum);
  end : example
endmodule
```

Output:

```
# KERNEL: Sum =   45
```

1.2. break and Continue (simple example)

Code:

```
module sample;
  int i;
  initial begin
    for (i = 0; i < 10; i++) begin
      if (i == 5) break;
      if (i % 2 == 0) continue;
      $display("Odd number = %0d", i);
    end
  end
endmodule
```

Output:

```
# KERNEL: Odd number = 1
# KERNEL: Odd number = 3
```

2. Tasks, Functions, and Void Functions

2.1. Functions

2.1.1 Example 1:addition of two number

Code:

```
module function_example;

  // Function definition
  function int add_numbers(int a, int b);
    add_numbers = a + b;  // Function returns sum of a and b
  endfunction

  initial begin
    int result;
    result = add_numbers(5, 3);   // Function call
    $display("Sum = %0d", result); // Output: Sum = 8
  end

endmodule
```

Output:

```
# KERNEL: Sum = 8
```

2.1.2 Example 2:finding largest of a number using function

```
    module max;//finding largest of a number using function

  function int max(int x ,int y);
    if(x<y)
      max=y;
    else
      max=x;
  endfunction

  initial begin
  int a=10;
  int b=20;
  int out;

  out=max(a,b);
    $display("The maximum value is = %0d", out);
  end
```

Output:

```
# KERNEL: The maximum value is = 20
```

Code:

```
module function_example;

  // Function definition
  function int add_numbers(int a, int b);
    add_numbers = a + b;  // Function returns sum of a and b
  endfunction

  initial begin
    int result;
    result = add_numbers(5, 3);    // Function call
    $display("Sum = %0d", result); // Output: Sum = 8
  end

endmodule
```

Output:

```
# KERNEL: Sum = 8
```

2.2. Tasks

2.2.1 Example 1 :static task

```
     module tb;

  initial display();
  initial display();
  initial display();
  initial display();

  // This is a static task
  task display();
    integer i = 0;
    i = i + 1;
    $display("i=%0d", i);
  endtask
endmodule
```

Output:

```
# KERNEL: i=1
# KERNEL: i=2
# KERNEL: i=3
# KERNEL: i=4
```

2.2.2 Example 2:automatic

```
     module tb;

  initial display();
  initial display();
  initial display();
  initial display();

  // Note that the task is now automatic
  task automatic display();
    integer i = 0;
    i = i + 1;
    $display("i=%0d", i);
  endtask
endmodule
```

Output:

```
i=1
i=1
i=1
i=1
```

2.3. Void Function

Example:

```
    module void_function_example;

  // Function definition
  function void say_hello(string name);
    $display("Hello, %s! Welcome to SystemVerilog!", name);
  endfunction

  initial begin
    say_hello("ABCD");   // Function call
    say_hello("EFGH");
  end

endmodule
```

output:

```
# KERNEL: Hello, ABCD! Welcome to SystemVerilog!
# KERNEL: Hello, EFGH! Welcome to SystemVerilog!
```