Experiment 2 – Fixed , Dynamic Arrays and Queues

Part 1:

Aim:

Write the code to design a D Flip Flop and write test bench in System Verilog.

Code:

DUT:

```
module dff(input d,clk, output reg q);
      always@(posedge clk) q<=d;
endmodule
```

TB:

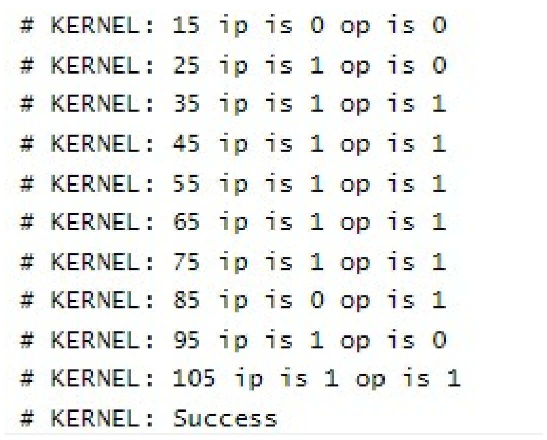
```
module tb();
  bit d,clk,q,y;
  dff duv(d,clk,q);
  int x = 0;

  initial forever #5 clk = !clk;

  initial begin
    repeat(10)begin
      @(negedge clk) begin
        d <= $random;y <= d;
      end

      @(posedge clk) if(q != y) x = x+1;
      $display("%0d ip is %d op is %d",$time,d,q);
    end
    if(!x) $display("Success"); else $display("Failure");
  end
endmodule
```

Output



```
# KERNEL: 15 ip is 0 op is 0
# KERNEL: 25 ip is 1 op is 0
# KERNEL: 35 ip is 1 op is 1
# KERNEL: 45 ip is 1 op is 1
# KERNEL: 55 ip is 1 op is 1
# KERNEL: 65 ip is 1 op is 1
# KERNEL: 75 ip is 1 op is 1
# KERNEL: 85 ip is 0 op is 1
# KERNEL: 95 ip is 1 op is 0
# KERNEL: 105 ip is 1 op is 1
# KERNEL: Success
```

Part 2:

Aim:

Write the code to design a 4 bit adder and write its test bench is System Verilog

Code:

DUT:

```
module add(input [3:0] a,b, input cin, output [3:0] sum, output cout);
  assign {cout,sum} = a+b+cin;
endmodule
```

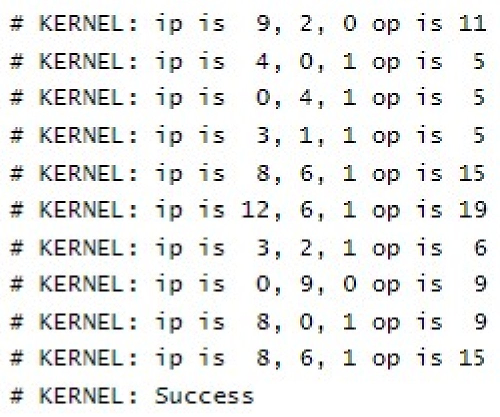
TB :

```
module tb();
  reg [3:0] a,b;
  wire [3:0] sum;
  reg cin;
  wire cout;
  int x=0;

  add duv(a,b,cin,sum,cout);

  initial begin
    repeat(10) begin
      {a,b,cin} = $random;
      #1;
      if({cout,sum} != a+b+cin) x = x+1;
      $display("ip is %d,%d, %b op is %d",a,b,cin,{cout,sum});
    end
    if(!x) $display("Success");
    else $display("Failure");
  end
endmodule
```

Output:



```
# KERNEL: ip is  9, 2, 0 op is 11
# KERNEL: ip is  4, 0, 1 op is  5
# KERNEL: ip is  0, 4, 1 op is  5
# KERNEL: ip is  3, 1, 1 op is  5
# KERNEL: ip is  8, 6, 1 op is 15
# KERNEL: ip is 12, 6, 1 op is 19
# KERNEL: ip is  3, 2, 1 op is  6
# KERNEL: ip is  0, 9, 0 op is  9
# KERNEL: ip is  8, 0, 1 op is  9
# KERNEL: ip is  8, 6, 1 op is 15
# KERNEL: Success
```

Result:

The given problem statement is executed and verified to be correct.