Abdelrahman Adw Ali

Digital Verification diploma SV8

Group 2 - No.1

FIFO Project.

1.bugs:

- wr_ack should be Low when reset is asserted .
- overflow should be Low when reset is asserted .
- overflow should be low when the fifo is not full.
- underflow should be Low when reset is asserted .
- underflow should be low when the fifo is not empty.
- underflow output should be Sequential .
- data_out should be low when the reset is asserted.
- Uncovered case when both wr_en and rd_en are high and FIFO if full , Reading process happens .
- Uncovered case when both wr_en and rd_en are high and FIFO if empty , Writing process happens .
- almostfull is high when there is two spots empty , while it should be when only one spot is empty

2.verification plan:

Label	Design Requirement Description	Stimulus Generation	Functional Coverage	Functionality Check
FIFO_1	When the reset is asserted the FIFO outputs (data_out & other flags) should be low .	Directed at the start of the simulation ,then randomized with constraint that drive reset to be off most of the time .	-	Immediate assertion to check for the async reset functionality
FIFO_2	When the wr_en is asserted, the FIFO is not Full , Writing Operation takes place withl data_in input , wr_ack should be activatd .	Randomization under constrains on the wr_en signal to be on with value(WR_EN_ON_DIST) of the time .	Cross Coverage betweenf wr_en with rd_en and wr_ack .	Concurrent assertion to check the wr_ack Functionality .
FIFO_3	When the FIFO is full , the full flag should be activated .	Randomization under constrains on the wr_en signal to be on with value(WR_EN_ON_DIST) of the time .	Cross Coverage betweenf wr_en with rd_en and full .	Immediate assertion to check for the full Flag functionality
FIFO_4	When the FIFO is empty , the empty flag should be activated .	Randomization under constrains on the rd_en signal to be on with value(RD_EN_ON_DIST) of the time .	Cross Coverage betweenf wr_en with rd_en and empty .	Immediate assertion to check for the empty Flag functionality
FIFO_5	When the FIFO has only 1 space left , the almostfull flag should be activated .	Randomization under constrains on the wr_en signal to be on with value(WR_EN_ON_DIST) of the time .	Cross Coverage betweenf wr_en with rd_en and almostfull .	Immediate assertion to check for the almostfull Flag functionality
FIFO_6	When the FIFO has only 1 space occupied , the almostempty flag should be activated .	Randomization under constrains on the rd_en signal to be on with value(RD_EN_ON_DIST) of the time .	Cross Coverage betweenf wr_en with rd_en and almostempty .	Immediate assertion to check for the almostempty Flag functionality
FIFO_7	When the FIFO is empty and the rd_en signal is high , underflow signal should be activated	Randomization under constrains on the rd_en signal to be on with value(RD_EN_ON_DIST) of the time .	Cross Coverage betweenf wr_en with rd_en and underflow .	Immediate assertion to check for the underflow Flag functionality
FIFO_8	When the FIFO is full and the wr_en signal is high , overflow signal should be activated	Randomization under constrains on the wr_en signal to be on with value(WR_EN_ON_DIST) of the time .	Cross Coverage betweenf wr_en with rd_en and overflow .	Immediate assertion to check for the overflow Flag functionality
FIFO_9	When the FIFO is not empty & rd_en signal is high the data_out should take the value of the FIFO element with the rd_ptr address and the rd_ptr is incremented .	Randomization under constrains on the rd_en signal to be on with value(RD_EN_ON_DIST) of the time .	-	Concurrent assertion to check the rd_ptr_wraparound & rd_ptr_theshold , Also a Reference model is made to check data_out Functionality .

Design code with assertions:

D: > Verification Diploma En.Kareem Waseem > Projects > Sync FIFO > FIFO.sv > FIFO

```
1  ////////////////////////////////////////////////////////////////////////////////
2  // Author: Kareem Waseem
3  // Course: Digital Verification using SV & UVM
4  //
5  // Description: FIFO Design
6  //
7  ////////////////////////////////////////////////////////////////////////////////
8  module FIFO(fifo_if.dut fifoif);
9
10     localparam max_fifo_addr = $clog2(fifoif.FIFO_DEPTH);
11
12     reg [fifoif.FIFO_WIDTH-1:0] mem [fifoif.FIFO_DEPTH-1:0];
13
14     reg [max_fifo_addr-1:0] wr_ptr, rd_ptr;
15     reg [max_fifo_addr:0] count;
16
17     always @(posedge fifoif.clk or negedge fifoif.rst_n) begin
18        if (!fifoif.rst_n) begin
19           wr_ptr <= 0;
20           // BUG DETECTED : wr_ack should be Low when reset is asserted.
21           fifoif.wr_ack <= 0 ;
22           // BUG DETECTED : overflow should be Low when reset is asserted.
23           fifoif.overflow <= 0 ;
24        end
25        else if (fifoif.wr_en && count < fifoif.FIFO_DEPTH) begin
26           mem[wr_ptr] <= fifoif.data_in;
27           fifoif.wr_ack <= 1;
28           // BUG DETECTED : overflow should be low when the fifo is not full.
29           fifoif.overflow <= 0;
30           wr_ptr <= wr_ptr + 1;
31        end
32        else begin
33           fifoif.wr_ack <= 0;
34           if (fifoif.full && fifoif.wr_en)
35              fifoif.overflow <= 1;
36           else
37              fifoif.overflow <= 0;
38        end
39     end
```

```
41  always @(posedge fifoif.clk or negedge fifoif.rst_n) begin
42      if (!fifoif.rst_n) begin
43          rd_ptr <= 0;
44          fifoif.underflow <= 0;
45          // BUG DETECTED : data_out should be low when the reset is asserted.
46          fifoif.data_out <=0;
47      end
48      else if (fifoif.rd_en && count != 0) begin
49          fifoif.data_out <= mem[rd_ptr];
50          // BUG DETECTED : overflow should be low when the fifo is not empty.
51          fifoif.underflow <= 0;
52          rd_ptr <= rd_ptr + 1;
53      end
54      // BUG DETECTED : Underflow output should be Sequential.
55      else begin
56          if (fifoif.empty & fifoif.rd_en)
57              fifoif.underflow <= 1;
58          else
59              fifoif.underflow <= 0;
60      end
61  end
62
63  always @(posedge fifoif.clk or negedge fifoif.rst_n) begin
64      if (!fifoif.rst_n) begin
65          count <= 0;
66      end
67      else begin
68          if ( ({fifoif.wr_en, fifoif.rd_en} == 2'b10) && !fifoif.full)
69              count <= count + 1;
70          else if ( ({fifoif.wr_en, fifoif.rd_en} == 2'b01) && !fifoif.empty)
71              count <= count - 1;
72          // BUG DETECTED : Uncovered case when both wr_en and rd_en are high and FIFO if full , Reading process happens .
73          else if ( ({fifoif.wr_en, fifoif.rd_en} == 2'b11) && fifoif.full)
74              count <= count - 1;
75          //BUG DETECTED : Uncovered case when both wr_en and rd_en are high and FIFO if empty , Writing process happens .
76          else if ( ({fifoif.wr_en, fifoif.rd_en} == 2'b11) && fifoif.empty)
77              count <= count + 1;
78
79      end
80  end
81
```

```
82  assign fifoif.full = (count == fifoif.FIFO_DEPTH)? 1 : 0;
83  assign fifoif.empty = (count == 0)? 1 : 0;
84  //BUG DETECTED : almostfull is high when there is two spots empty , while it should be only one .
85  assign fifoif.almostfull = (count == fifoif.FIFO_DEPTH-1)? 1 : 0;
86  assign fifoif.almostempty = (count == 1)? 1 : 0;
87
88
89  `ifdef SIM
90      //////////////////////////////////////// assertions //////////////////////////////////////////////
91      always_comb begin
92          if(!fifoif.rst_n)begin
93              reset_wr_ptr: assert final (wr_ptr == 0);
94              reset_rd_ptr: assert final (rd_ptr == 0);
95              reset_count : assert final (count == 0);
96          end
97      end
98
99      property wr_ack_p;
100         @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
101             ( fifoif.wr_en && count < fifoif.FIFO_DEPTH ) |=>
102                 (fifoif.wr_ack == 1);
103     endproperty
104     a_wr_ack_p: assert property ( property wr_ack_p; ("error in wr_ack");
105     c_wr_ack_p: cover  property (wr_ack_p);
106
```

```
106     property overflow_p;
107        @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
108            ( fifoif.wr_en && fifoif.full ) |=>
109                (fifoif.overflow == 1);
110     endproperty
111     a_overflow_p: assert property (overflow_p) else $error("error in overflow");
112     c_overflow_p: cover  property (overflow_p);
113
114     property underflow_p;
115         @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
116             ( fifoif.rd_en && fifoif.empty ) |=>
117                (fifoif.underflow == 1);
118     endproperty
119     a_underflow_p: assert property (underflow_p) else $error("error in underflow");
120     c_underflow_p: cover  property (underflow_p);
121
122     property empty_p;
123         @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
124             ( count == 0 ) |->
125                (fifoif.empty);
126      endproperty
127      a_empty_p: assert property (empty_p) else $error("error in empty");
128      c_empty_p: cover  property (empty_p);
129
130      property full_p;
131         @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
132            ( count == fifoif.FIFO_DEPTH ) |->
133               (fifoif.full);
134       endproperty
135       a_full_p: assert property (full_p) else $error("error in full");
136       c_full_p: cover  property (full_p);
137
138      property almostfull_p;
139         @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
140            ( count == fifoif.FIFO_DEPTH - 1) |->
141                (fifoif.almostfull);
142      endproperty
143      a_almostfull_p: assert property (almostfull_p) else $error("error in almostfull");
144      c_almostfull_p: cover  property (almostfull_p);
```

```
146     property almostempty_p;
147       @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
148           ( count == 1 ) |->
149             (fifoif.almostempty);
150     endproperty
151     a_almostempty_p: assert property (almostempty_p) else $error("error in almostempty");
152     c_almostempty_p: cover  property (almostempty_p);
153
154    property wr_ptr_wrap_around_p;
155       @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
156           ( wr_ptr == 7 ) |-> (wr_ptr == 0) [=1];
157     endproperty
158     a_wr_ptr_wrap_around_p: assert property (wr_ptr_wrap_around_p) else $error("error in wr_ptr_wrap_around");
159     c_wr_ptr_wrap_around_p: cover  property (wr_ptr_wrap_around_p);
160
161     property rd_ptr_wrap_around_p;
162       @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
163         ( rd_ptr == 7 ) |-> (rd_ptr == 0) [=1];
164     endproperty
165     a_rd_ptr_wrap_around_p: assert property (rd_ptr_wrap_around_p) else $error("error in rd_ptr_wrap_around");
166     c_rd_ptr_wrap_around_p: cover  property (rd_ptr_wrap_around_p);
167
168     property count_p;
169       @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
170           ( count == 8 ) |-> (count == 0 || count == 7) [->1] ;
171     endproperty
172     a_count_p: assert property (count_p) else $error("error in count");
173     c_count_p: cover  property (count_p);
174
175     property wr_operation_p;
176        @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
177           ( (fifoif.wr_en && count < fifoif.FIFO_DEPTH) )
178            |=> ( mem[$past(wr_ptr)] == $past(fifoif.data_in) );
179     endproperty
180     a_wr_operation_p: assert property (wr_operation_p) else $error("error in wr_operation");
181     c_wr_operation_p: cover  property (wr_operation_p);
182
183     property check12 ;
184      @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
185           (wr_ptr < fifoif.FIFO_DEPTH) ;
186      endproperty
187     assert property (check12 ) else $error("error in check12 ");
188     cover  property (check12 );
```

```
189         property check13 ;
190             @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
191                 (rd_ptr < fifoif.FIFO_DEPTH) ;
192         endproperty
193         assert property (check13) else $error("error in check13");
194         cover  property (check13);
195 
196         property check14 ;
197             @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
198                 (count <= fifoif.FIFO_DEPTH) ;
199         endproperty
200         assert property (check14) else $error("error in check14");
201         cover  property (check14);
202 
203         property check15;
204             @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
205                 ( ({fifoif.wr_en, fifoif.rd_en} == 2'b10) && !fifoif.full) || ( ({fifoif.wr_en, fifoif.rd_en} == 2'b11) && fifoif.empty)
206                     |=> (count == $past(count)+1);
207         endproperty
208         assert property (check15) else $error("error in check15");
209         cover  property (check15);
210 
211         property check16;
212             @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
213                 ( ({fifoif.wr_en, fifoif.rd_en} == 2'b01) && !fifoif.empty) || ( ({fifoif.wr_en, fifoif.rd_en} == 2'b11) && fifoif.full)
214                     |=> (count == $past(count)-1);
215         endproperty
216         assert property (check16) else $error("error in check16");
217         cover  property (check16);
218 
219         property check17;
220             @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
221                 ( count <fifoif.FIFO_DEPTH -1)
222                     |-> (!fifoif.almostfull);
223         endproperty
224         assert property (check17) else $error("error in check17");
225         cover  property (check17);
226 
227         property check18;
228             @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
229                 !(fifoif.overflow && fifoif.wr_ack);
230         endproperty
231         assert property (check18) else $error("error in check18");
232         cover  property (check18);
233     `endif
234 
235 endmodule
```

3.interface code:

D: > Verification Diploma En.Kareem Waseem > Projects > Sync FIFO > fifo_if.sv > ...

```
1   interface fifo_if(input bit clk);
2       parameter FIFO_WIDTH = 16;
3       parameter FIFO_DEPTH = 8;
4       logic [FIFO_WIDTH-1:0] data_in;
5       logic rst_n;
6       logic wr_en;
7       logic rd_en;
8       logic  [FIFO_WIDTH-1:0] data_out;
9       logic  wr_ack;
10      logic overflow;
11      logic full, empty, almostfull, almostempty, underflow;
12      modport dut(
13          input data_in, wr_en, rd_en, clk, rst_n,
14          output full, empty, almostfull, almostempty, wr_ack, overflow, underflow, data_out
15      );
16      modport test(
17          output data_in, wr_en, rd_en, rst_n,
18          input clk,full, empty, almostfull, almostempty, wr_ack, overflow, underflow, data_out
19      );
20      modport monitor(
21          input data_in, wr_en, rd_en, rst_n,clk,full, empty, almostfull, almostempty, wr_ack, overflow, underflow, data_out
22      );
23  
24  
25  endinterface
```

4.shared_pkg:

```
package shared_pkg;

    int error_count,correct_count;
    bit test_finished;

endpackage
```

5. transaction_pkg:

D: > Verification Diploma En.Kareem Waseem > Projects > Sync FIFO > fifo_transaction.sv > ...

```
package fifo_transaction_pkg;
    parameter FIFO_WIDTH = 16;
    parameter FIFO_DEPTH = 8;
    class fifo_transaction #(parameter FIFO_WIDTH = 16,
                             parameter FIFO_DEPTH = 8);

        rand logic [FIFO_WIDTH-1:0] data_in;
        rand logic rst_n;
        rand logic wr_en;
        rand logic rd_en;

        logic  [FIFO_WIDTH-1:0] data_out;
        logic  wr_ack;
        logic overflow;
        logic full, empty, almostfull, almostempty, underflow;
        int RD_EN_ON_DIST, WR_EN_ON_DIST;

        function new(int RD_EN_ON_DIST = 30, int WR_EN_ON_DIST = 70);
            this.RD_EN_ON_DIST = RD_EN_ON_DIST;
            this.WR_EN_ON_DIST = WR_EN_ON_DIST;
        endfunction

        constraint res_cs{
            rst_n dist { 1 :/98, 0 :/2};
        }
        constraint wr_en_cs{
            wr_en dist { 1 :/WR_EN_ON_DIST, 0 :/ 100-WR_EN_ON_DIST};
        }
        constraint rd_en_cs{
            rd_en dist { 1 :/RD_EN_ON_DIST, 0 :/ 100-RD_EN_ON_DIST};
        }

    endclass
endpackage
```

6.scoreboard_pkg:

```
package fifo_scoreboard_pkg;

    import fifo_transaction_pkg::*;
    import shared_pkg::*;

    class fifo_scoreboard;

        parameter FIFO_WIDTH = 16;
        parameter FIFO_DEPTH = 8;

        logic  [FIFO_WIDTH-1:0] data_out_ref;
        logic  wr_ack_ref;
        logic overflow_ref;
        logic full_ref, empty_ref, almostfull_ref, almostempty_ref, underflow_ref;
        localparam max_fifo_addr = $clog2(FIFO_DEPTH);
        logic [FIFO_WIDTH-1:0] queue[$];
        logic [max_fifo_addr:0] counter;

        function void check_data(fifo_transaction F_txn);
            logic [6:0]ref_flags,dut_flags;
            reference_model(F_txn);
            ref_flags = {wr_ack_ref, overflow_ref, full_ref, empty_ref, almostfull_ref, almostempty_ref, underflow_ref};
            dut_flags = {F_txn.wr_ack, F_txn.overflow, F_txn.full, F_txn.empty, F_txn.almostfull,  F_txn.almostempty, F_txn.underflow};
            if( (F_txn.data_out == data_out_ref) &&(dut_flags == ref_flags))begin
                correct_count++;
            end
            else begin
                error_count++;
                $display("there is an error @ time : %0t",$time);
                $display("dut_inputs: rst_n = %0b, wr_en = %0b, rd_en = %0b",F_txn.rst_n, F_txn.wr_en, F_txn.rd_en);
                $display(" dut_flags : {wr_ack,overflow,full,empty,almostfull,almostempty,underflow}=0b%7b",dut_flags);
                $display(" ref_flags : {wr_ack,overflow,full,empty,almostfull,almostempty,underflow}=0b%7b",ref_flags);
                $display("  dut_out  : %0h, ref_out : %0h",F_txn.data_out, data_out_ref);
            end
        endfunction
        function void reference_model(fifo_transaction F_ref_txn);
            //write check
            if(!F_ref_txn.rst_n)begin
                wr_ack_ref = 0;
                overflow_ref =0;
                data_out_ref = 0;
                underflow_ref =0;
                queue.delete();
            end
```

```
45              else begin
46                  if(F_ref_txn.wr_en && (counter < FIFO_DEPTH))begin
47                      queue.push_back(F_ref_txn.data_in);
48                      wr_ack_ref = 1;
49                      overflow_ref =0;
50                  end
51                  else begin
52                      wr_ack_ref =0;
53                      if(F_ref_txn.wr_en && full_ref)begin
54                          overflow_ref =1;
55                      end
56                      else begin
57                          overflow_ref =0;
58                      end
59                  end
60                  if(F_ref_txn.rd_en && (counter != 0))begin
61                      data_out_ref = queue.pop_front();
62                      underflow_ref =0;
63                  end
64                  else begin
65                      if(F_ref_txn.rd_en && empty_ref)begin
66                          underflow_ref =1;
67                      end
68                      else begin
69                          underflow_ref =0;
70                      end
71                  end
72              end
73              if (!F_ref_txn.rst_n) begin
74                counter = 0;
75              end else begin
76                case ({F_ref_txn.wr_en, F_ref_txn.rd_en})
77                  2'b10: if (!full_ref)  counter = counter + 1;
78                  2'b01: if (!empty_ref) counter = counter - 1;
79                  2'b11: begin
80                    if (full_ref && !empty_ref) counter = counter - 1; // writing when full, read takes priority
81                    else if (empty_ref && !full_ref) counter = counter + 1;
82                  end
83                endcase
84              end
85              full_ref = (counter == FIFO_DEPTH)?1:0;
86              empty_ref = (counter == 0)?1:0;
87              almostfull_ref  = (counter == FIFO_DEPTH - 1)?1:0;
88              almostempty_ref = (counter == 1)?1:0;
89          endfunction
90      endclass
91  endpackage
```

7. coverage_pkg:

```
package fifo_coverage_pkg;

    import fifo_transaction_pkg::*;

    class fifo_coverage;
        fifo_transaction fifo_cvg_txn;
        covergroup fifo_cg;
            wr_en_cp : coverpoint fifo_cvg_txn.wr_en{
                bins wr_en_on = {1};
                bins wr_en_off = {0};
            }
            rd_en_cp : coverpoint fifo_cvg_txn.rd_en{
                bins rd_en_on = {1};
                bins rd_en_off = {0};
            }
            wr_ack_cp : coverpoint fifo_cvg_txn.wr_ack{
                bins wr_ack_on = {1};
                bins wr_ack_off = {0};
            }
            overflow_cp : coverpoint fifo_cvg_txn.overflow{
                bins overflow_on = {1};
                bins overflow_off = {0};
            }
            full_cp : coverpoint fifo_cvg_txn.full{
                bins full_on = {1};
                bins full_off = {0};
            }
            empty_cp : coverpoint fifo_cvg_txn.empty{
                bins empty_on = {1};
                bins empty_off = {0};
            }
            almostfull_cp : coverpoint fifo_cvg_txn.almostfull{
                bins almostfull_on = {1};
                bins almostfull_off = {0};
            }
            almostempty_cp : coverpoint fifo_cvg_txn.almostempty{
                bins almostempty_on = {1};
                bins almostempty_off = {0};
            }
            underflow_cp : coverpoint fifo_cvg_txn.underflow{
                bins underflow_on = {1};
                bins underflow_off = {0};
            }
            wr_rd_ack_cp : cross wr_en_cp, rd_en_cp, wr_ack_cp{
                illegal_bins wr_en_ack_on = binsof(wr_en_cp) intersect {0} && binsof(wr_ack_cp) intersect {1};
            }
            wr_rd_overflow_cp: cross wr_en_cp, rd_en_cp, overflow_cp{
                illegal_bins wr_en_overflow_on = binsof(wr_en_cp) intersect {0} && binsof(overflow_cp) intersect {1};
            }
            wr_rd_full_cp: cross wr_en_cp, rd_en_cp, full_cp{
                illegal_bins wr_en_overflow_on = binsof(rd_en_cp) intersect {1} && binsof(full_cp) intersect {1};
            }
            wr_rd_empty_cp: cross wr_en_cp, rd_en_cp, empty_cp;
            wr_rd_almostfull_cp: cross wr_en_cp, rd_en_cp, almostfull_cp;
            wr_rd_almostempty_cp: cross wr_en_cp, rd_en_cp, almostempty_cp;
            wr_rd_underflow_cp: cross wr_en_cp, rd_en_cp, underflow_cp{
                illegal_bins wr_en_overflow_on = binsof(rd_en_cp) intersect {0} && binsof(underflow_cp) intersect {1};
            }
        endgroup

        function new;
            fifo_cg = new;
        endfunction

        function void sample_data(fifo_transaction F_txn);
            fifo_cvg_txn = F_txn;
            fifo_cg.sample();
        endfunction
    endclass
endpackage
```

8.monitor code:

```
import fifo_scoreboard_pkg ::*;
import fifo_coverage_pkg ::*;
import shared_pkg ::*;

module fifo_monitor (fifo_if.monitor fifoif);

  fifo_transaction FIFO_mon_txn  =new();
  fifo_scoreboard FIFO_mon_sb =new();
  fifo_coverage FIFO_mon_cov = new();
  initial begin

      forever begin
        @(negedge fifoif.clk);

        FIFO_mon_txn.rst_n       = fifoif.rst_n;
        FIFO_mon_txn.wr_en       = fifoif.wr_en;
        FIFO_mon_txn.rd_en       = fifoif.rd_en;
        FIFO_mon_txn.data_in     = fifoif.data_in;
        FIFO_mon_txn.data_out    = fifoif.data_out;
        FIFO_mon_txn.wr_ack      = fifoif.wr_ack;
        FIFO_mon_txn.overflow    = fifoif.overflow;
        FIFO_mon_txn.full        = fifoif.full;
        FIFO_mon_txn.empty       = fifoif.empty;
        FIFO_mon_txn.almostfull  = fifoif.almostfull;
        FIFO_mon_txn.almostempty = fifoif.almostempty;
        FIFO_mon_txn.underflow   = fifoif.underflow;
        fork
            begin
              @(posedge fifoif.clk);
              FIFO_mon_cov.sample_data(FIFO_mon_txn);
            end
            begin
              @(posedge fifoif.clk);
              FIFO_mon_sb.check_data(FIFO_mon_txn);
            end
        join
        if(test_finished) begin
            $display("Simulation Stopped : Error count = %0d , Correct count = %0d",error_count,correct_count);
            $stop;
        end
      end

  end

endmodule
```

9. test code:

D: > Verification Diploma En.Kareem Waseem > Projects > Sync FIFO > fifo_test.

```
1   import fifo_transaction_pkg ::*;
2   import shared_pkg::*;
3
4   module fifo_test (fifo_if.test fifoif);
5
6         fifo_transaction test_txn = new();
7
8         initial begin
9               fifoif.rst_n = 0 ;
10              repeat(2) @(negedge fifoif.clk);
11              repeat(2000) begin
12                    @(negedge fifoif.clk);
13                    assert(test_txn.randomize());
14                    fifoif.rst_n = test_txn.rst_n ;
15                    fifoif.wr_en = test_txn.wr_en ;
16                    fifoif.rd_en = test_txn.rd_en ;
17                    fifoif.data_in = test_txn.data_in ;
18              end
19              test_finished = 1 ;
20        end
21  endmodule
```

10. top code:

D: > Verification Diploma En.Kareem Waseem > Projects > Sync FIFO

```
1   module fifo_top();
2
3       bit clk;
4       initial begin
5           clk = 0;
6           forever begin
7               #10 clk = ~clk;
8           end
9       end
10
11      fifo_if fifoif(clk);
12      FIFO DUT(fifoif);
13      fifo_test test(fifoif);
14      fifo_monitor monitor(fifoif);
15  endmodule
```

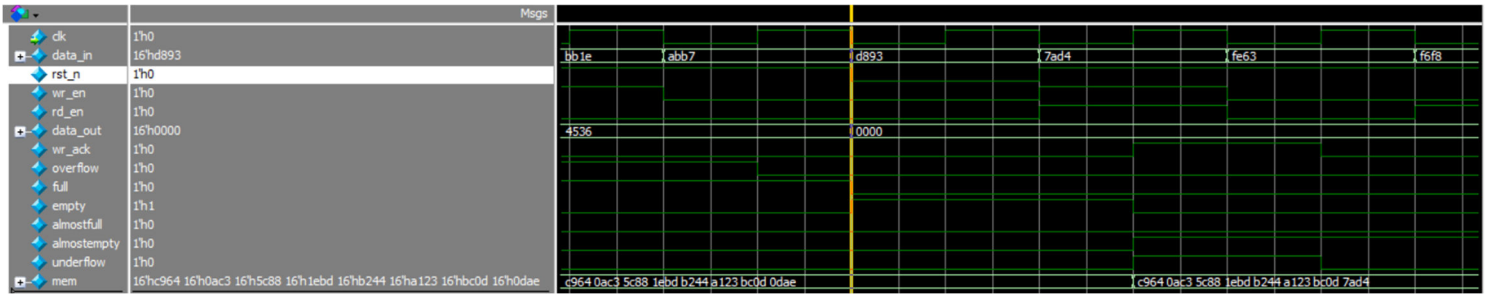
Do file:

D: > Verification Diploma En.Kareem Waseem > Projects > Sync FIFO > run.do

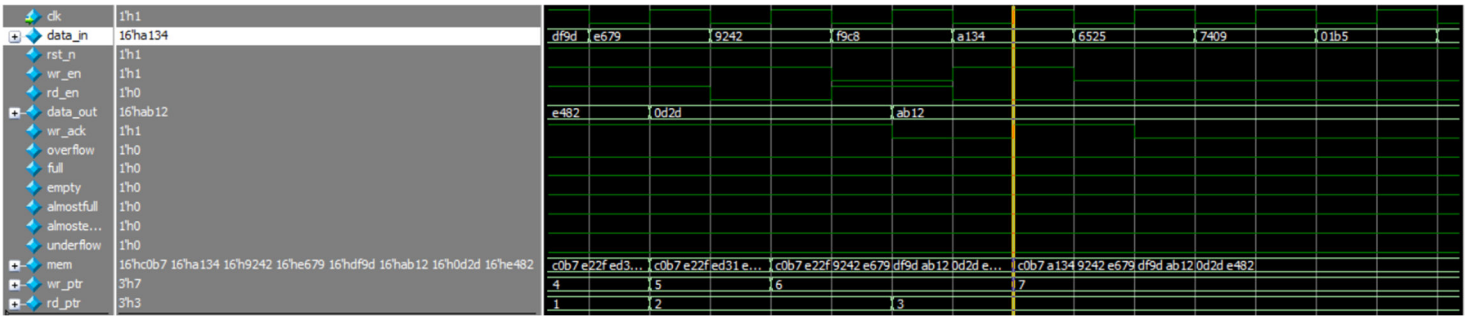
```
vlib work
vlog -f fifo_files.list +cover -covercells +define+SIM
vsim -voptargs=+acc work.fifo_top -cover
add wave /fifo_top/fifoif/*
add wave -position insertpoint \
/fifo_top/DUT/mem \
/fifo_top/DUT/wr_ptr \
/fifo_top/DUT/rd_ptr \
/fifo_top/DUT/count
coverage save fifo.ucdb -onexit
run -all
# vcover report fifo.ucdb -details -annotate -all -output coverage_rpt.txt
```

Questa snippets:

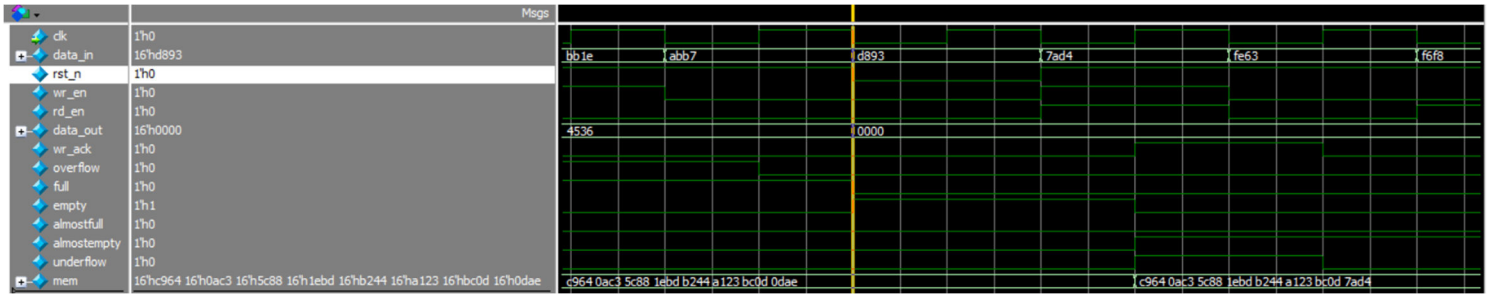
FIFO_1: (When the reset is asserted the FIFO outputs should be low).



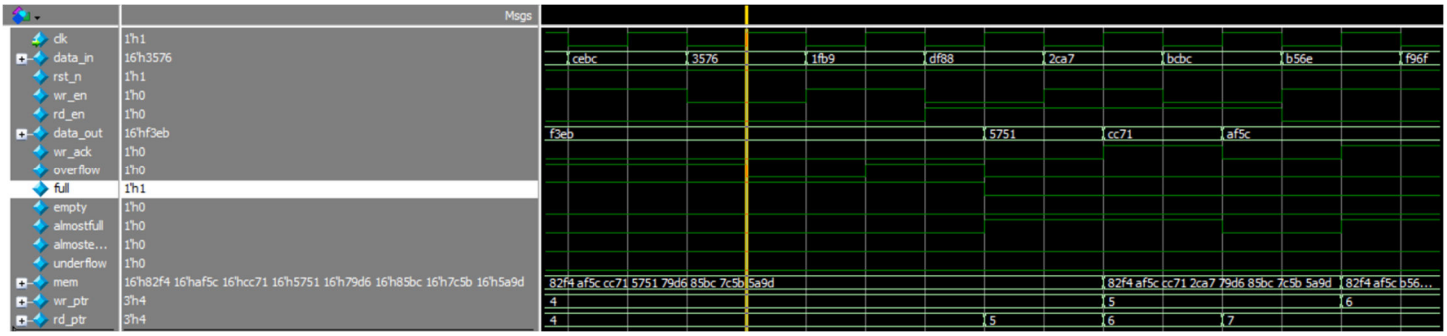
FIFO_2 : (When the wr_en is asserted, the FIFO is not Full , Writing Operation takes place with data_in input , wr_ack should be activated).



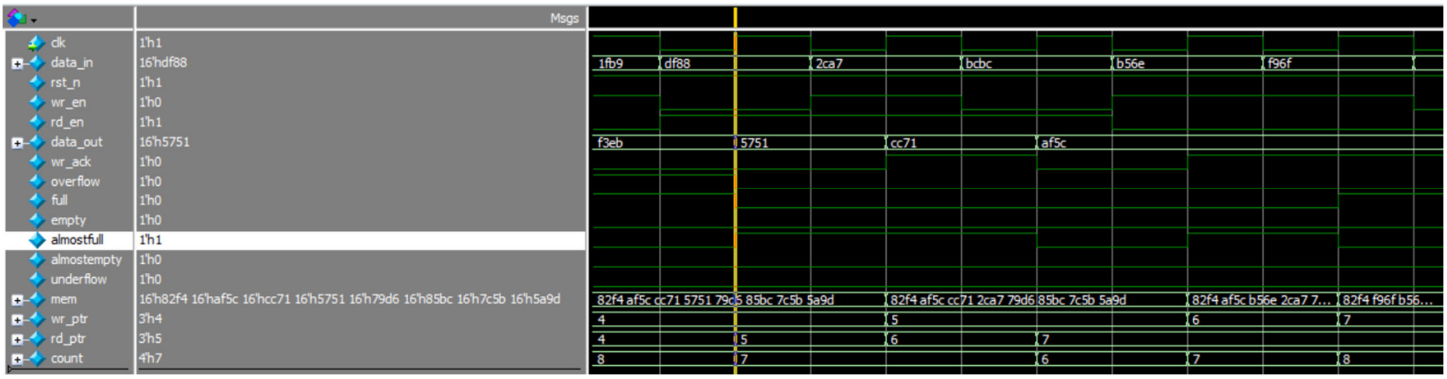
FIFO_3: (When the FIFO is full , the full flag should be activated).



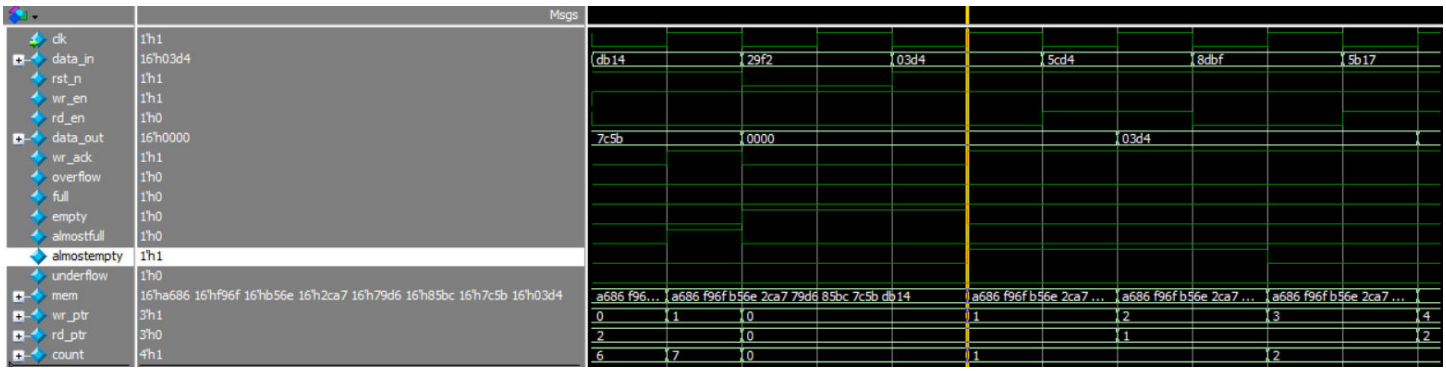
FIFO_4 : (When the FIFO is empty , the empty flag should be activated).



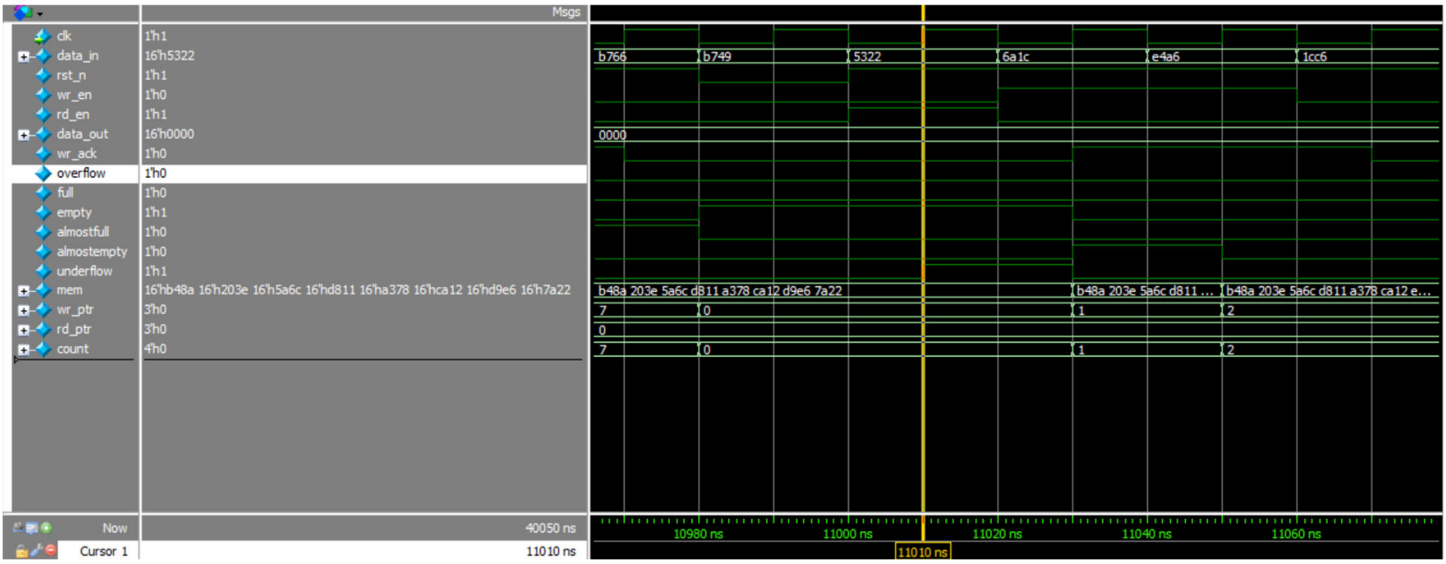
FIFO_5 : (When the FIFO has only 1 space left , the almostfull flag should be activated).



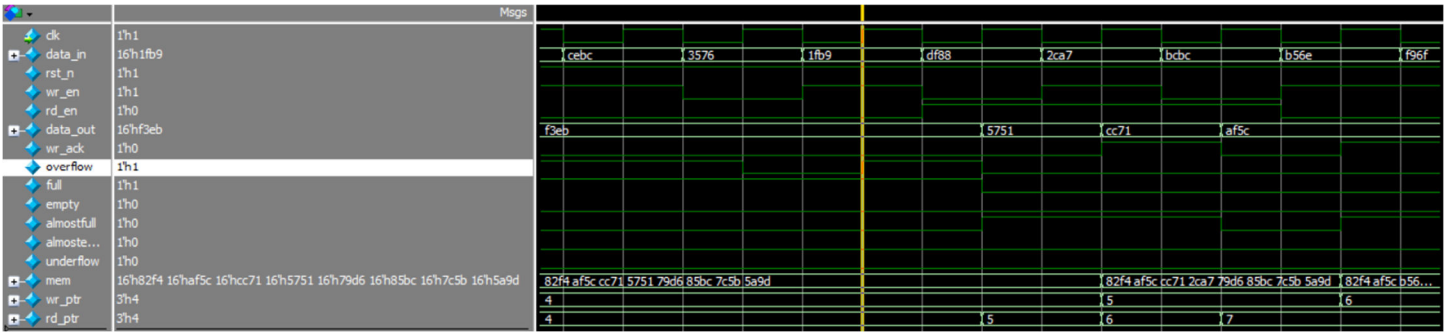
FIFO_6 : (When the FIFO has only 1 space occupied , the almostempty flag should be activated).



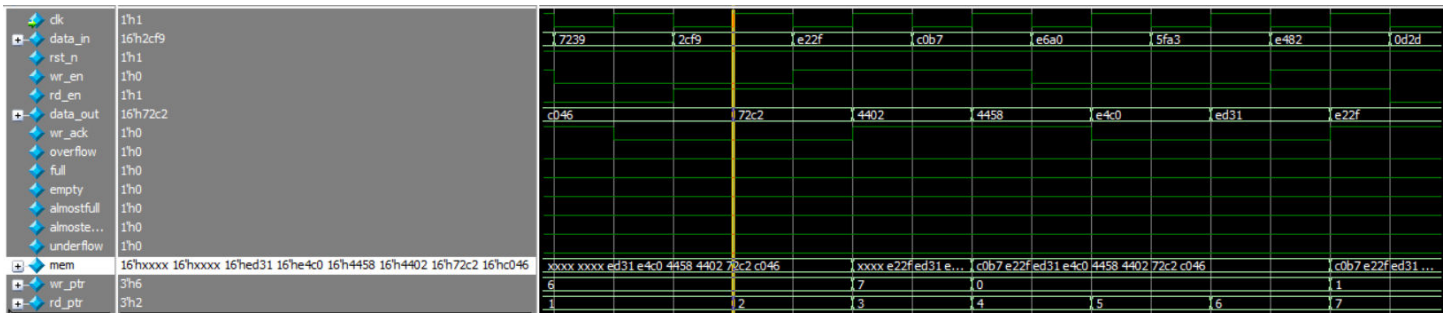
FIFO_7 : (When the FIFO is empty and the rd_en signal is high , underflow signal should be activated).



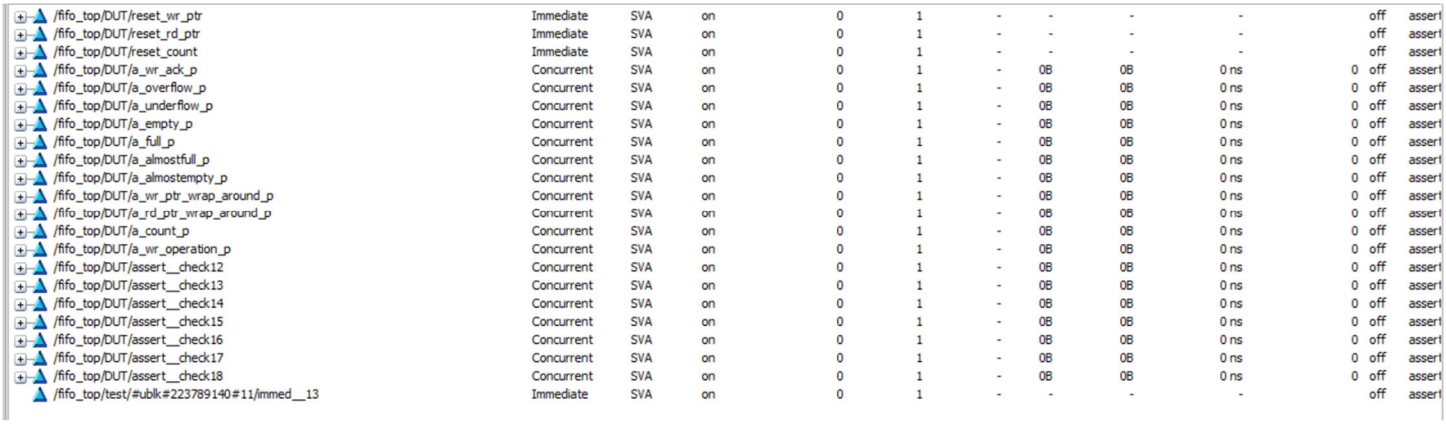
FIFO_8 : (When the FIFO is full and the wr_en signal is high , overflow signal should be activated).



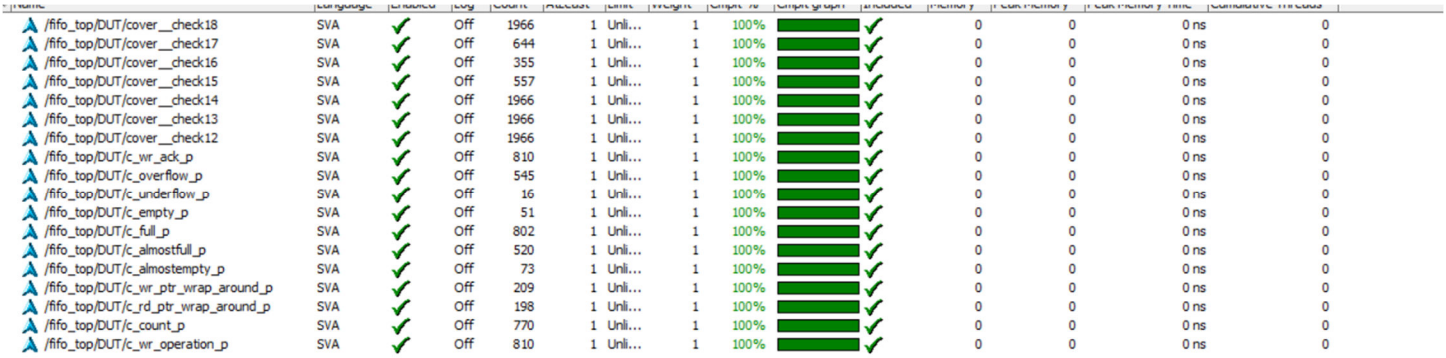
FIFO_9 : (When the FIFO is not empty & rd_en signal is high the data_out should take the value of the FIFO element with the rd_ptr address and the rd_ptr is incremented).



Assertions:

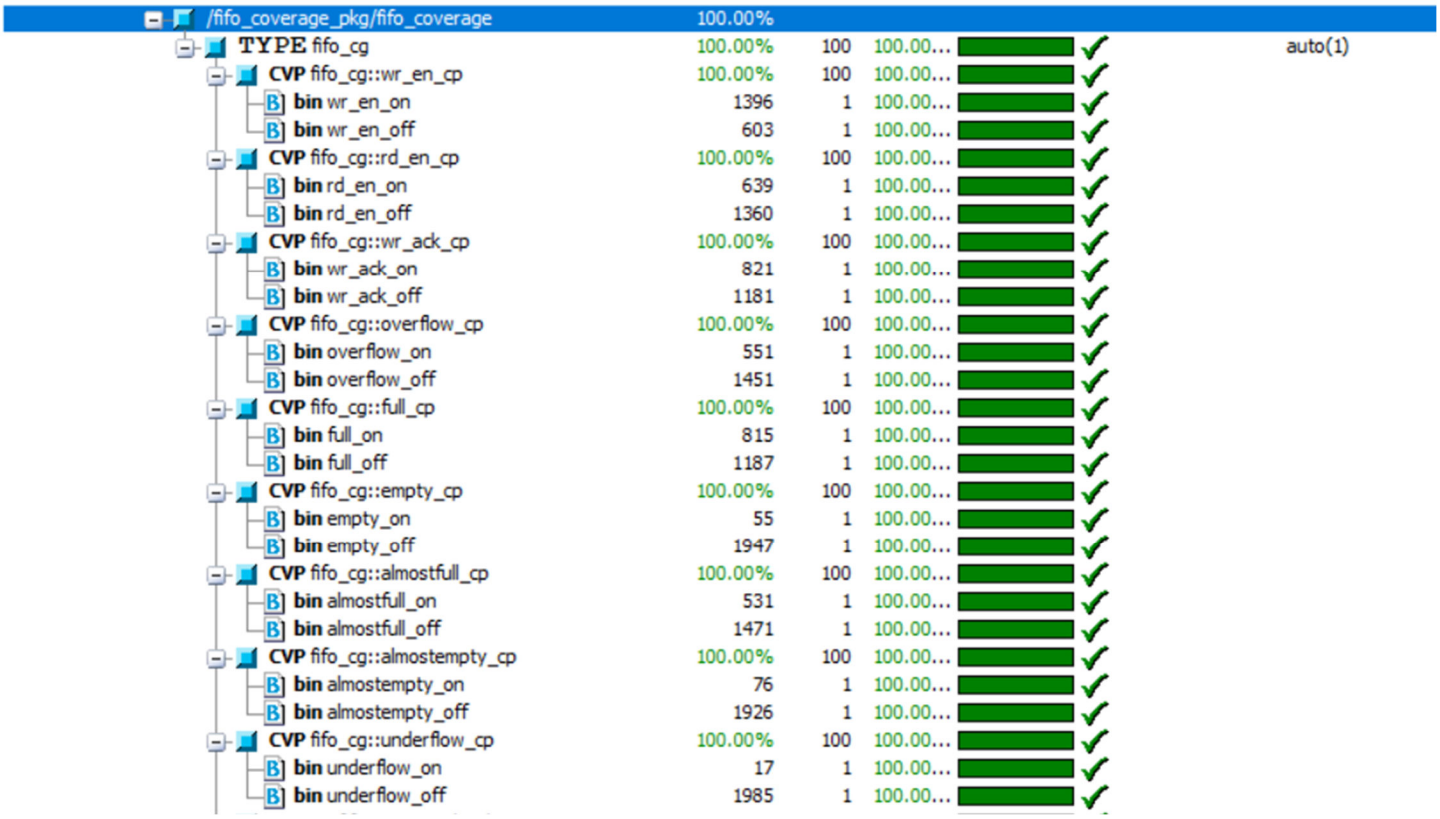


/fifo_top/DUT/reset_wr_ptr	Immediate	SVA	on	0	1	-	-	-	-		off	assert
/fifo_top/DUT/reset_rd_ptr	Immediate	SVA	on	0	1	-	-	-	-		off	assert
/fifo_top/DUT/reset_count	Immediate	SVA	on	0	1	-	-	-	-		off	assert
/fifo_top/DUT/a_wr_ack_p	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert
/fifo_top/DUT/a_overflow_p	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert
/fifo_top/DUT/a_underflow_p	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert
/fifo_top/DUT/a_empty_p	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert
/fifo_top/DUT/a_full_p	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert
/fifo_top/DUT/a_almostfull_p	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert
/fifo_top/DUT/a_almostempty_p	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert
/fifo_top/DUT/a_wr_ptr_wrap_around_p	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert
/fifo_top/DUT/a_rd_ptr_wrap_around_p	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert
/fifo_top/DUT/a_count_p	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert
/fifo_top/DUT/a_wr_operation_p	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert
/fifo_top/DUT/assert__check12	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert
/fifo_top/DUT/assert__check13	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert
/fifo_top/DUT/assert__check14	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert
/fifo_top/DUT/assert__check15	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert
/fifo_top/DUT/assert__check16	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert
/fifo_top/DUT/assert__check17	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert
/fifo_top/DUT/assert__check18	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert
/fifo_top/test/#ublk#223789140#11/immed__13	Immediate	SVA	on	0	1	-	-	-	-		off	assert

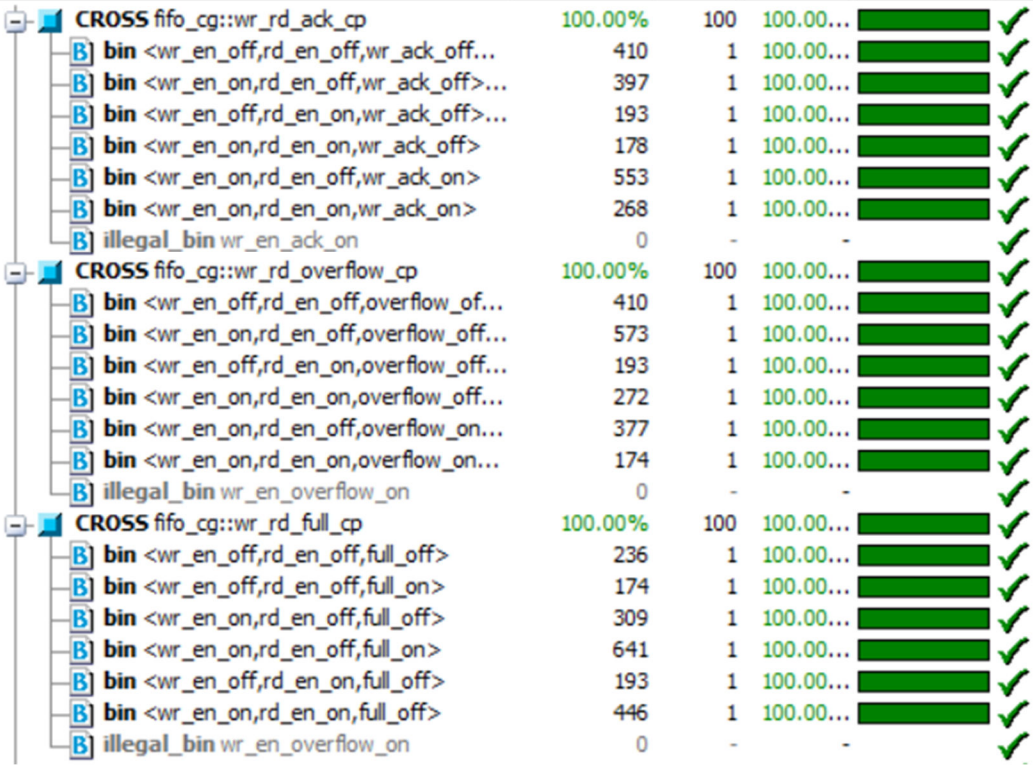


/fifo_top/DUT/cover__check18	SVA	✓	Off	1966	1	Unli...	1	100%	✓	0	0	0 ns	0
/fifo_top/DUT/cover__check17	SVA	✓	Off	644	1	Unli...	1	100%	✓	0	0	0 ns	0
/fifo_top/DUT/cover__check16	SVA	✓	Off	355	1	Unli...	1	100%	✓	0	0	0 ns	0
/fifo_top/DUT/cover__check15	SVA	✓	Off	557	1	Unli...	1	100%	✓	0	0	0 ns	0
/fifo_top/DUT/cover__check14	SVA	✓	Off	1966	1	Unli...	1	100%	✓	0	0	0 ns	0
/fifo_top/DUT/cover__check13	SVA	✓	Off	1966	1	Unli...	1	100%	✓	0	0	0 ns	0
/fifo_top/DUT/cover__check12	SVA	✓	Off	1966	1	Unli...	1	100%	✓	0	0	0 ns	0
/fifo_top/DUT/c_wr_ack_p	SVA	✓	Off	810	1	Unli...	1	100%	✓	0	0	0 ns	0
/fifo_top/DUT/c_overflow_p	SVA	✓	Off	545	1	Unli...	1	100%	✓	0	0	0 ns	0
/fifo_top/DUT/c_underflow_p	SVA	✓	Off	16	1	Unli...	1	100%	✓	0	0	0 ns	0
/fifo_top/DUT/c_empty_p	SVA	✓	Off	51	1	Unli...	1	100%	✓	0	0	0 ns	0
/fifo_top/DUT/c_full_p	SVA	✓	Off	802	1	Unli...	1	100%	✓	0	0	0 ns	0
/fifo_top/DUT/c_almostfull_p	SVA	✓	Off	520	1	Unli...	1	100%	✓	0	0	0 ns	0
/fifo_top/DUT/c_almostempty_p	SVA	✓	Off	73	1	Unli...	1	100%	✓	0	0	0 ns	0
/fifo_top/DUT/c_wr_ptr_wrap_around_p	SVA	✓	Off	209	1	Unli...	1	100%	✓	0	0	0 ns	0
/fifo_top/DUT/c_rd_ptr_wrap_around_p	SVA	✓	Off	198	1	Unli...	1	100%	✓	0	0	0 ns	0
/fifo_top/DUT/c_count_p	SVA	✓	Off	770	1	Unli...	1	100%	✓	0	0	0 ns	0
/fifo_top/DUT/c_wr_operation_p	SVA	✓	Off	810	1	Unli...	1	100%	✓	0	0	0 ns	0

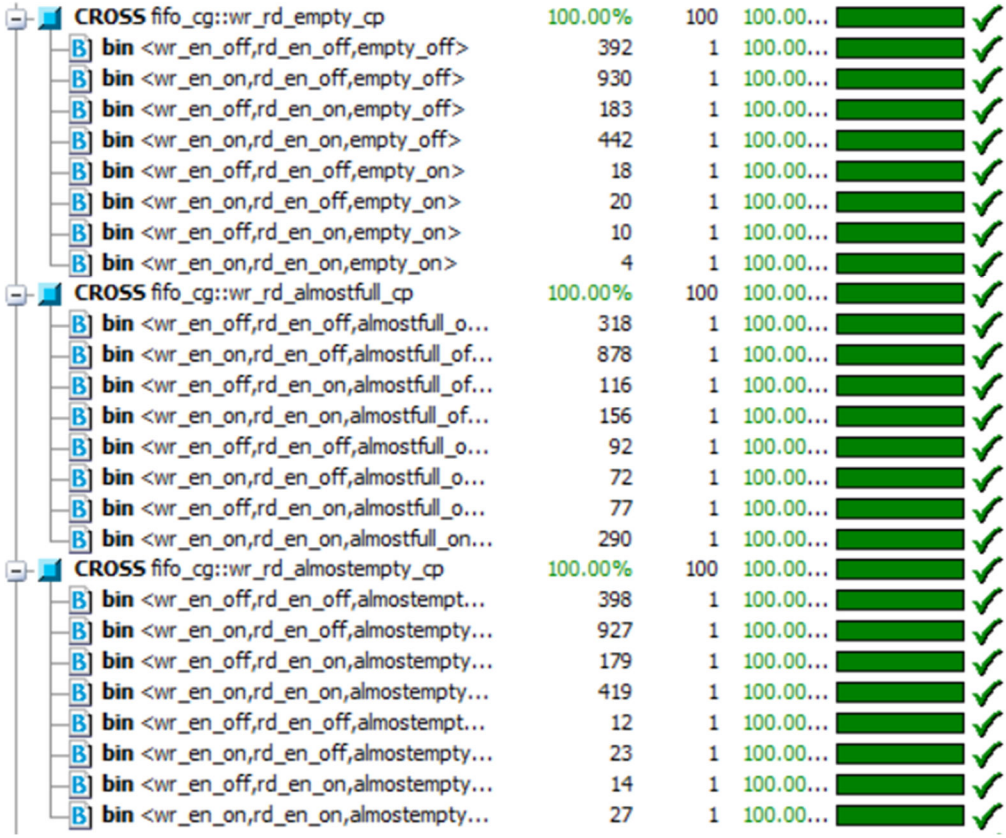
covergroups:



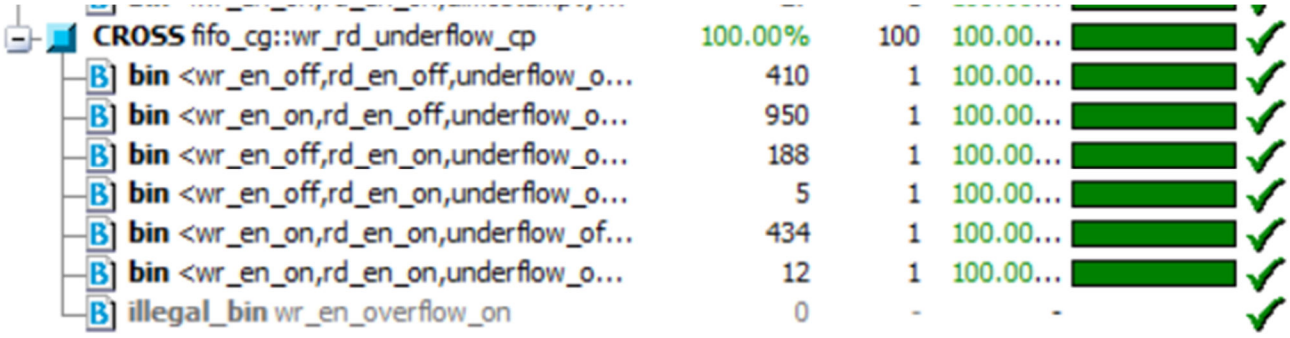
/fifo_coverage_pkg/fifo_coverage	100.00%			
TYPE fifo_cg	100.00%	100	100.00...	auto(1)
CVP fifo_cg::wr_en_cp	100.00%	100	100.00...	
bin wr_en_on	1396	1	100.00...	
bin wr_en_off	603	1	100.00...	
CVP fifo_cg::rd_en_cp	100.00%	100	100.00...	
bin rd_en_on	639	1	100.00...	
bin rd_en_off	1360	1	100.00...	
CVP fifo_cg::wr_ack_cp	100.00%	100	100.00...	
bin wr_ack_on	821	1	100.00...	
bin wr_ack_off	1181	1	100.00...	
CVP fifo_cg::overflow_cp	100.00%	100	100.00...	
bin overflow_on	551	1	100.00...	
bin overflow_off	1451	1	100.00...	
CVP fifo_cg::full_cp	100.00%	100	100.00...	
bin full_on	815	1	100.00...	
bin full_off	1187	1	100.00...	
CVP fifo_cg::empty_cp	100.00%	100	100.00...	
bin empty_on	55	1	100.00...	
bin empty_off	1947	1	100.00...	
CVP fifo_cg::almostfull_cp	100.00%	100	100.00...	
bin almostfull_on	531	1	100.00...	
bin almostfull_off	1471	1	100.00...	
CVP fifo_cg::almostempty_cp	100.00%	100	100.00...	
bin almostempty_on	76	1	100.00...	
bin almostempty_off	1926	1	100.00...	
CVP fifo_cg::underflow_cp	100.00%	100	100.00...	
bin underflow_on	17	1	100.00...	
bin underflow_off	1985	1	100.00...	



CROSS fifo_cg::wr_rd_ack_cp	100.00%	100	100.00...	
bin <wr_en_off,rd_en_off,wr_ack_off...	410	1	100.00...	
bin <wr_en_on,rd_en_off,wr_ack_off>...	397	1	100.00...	
bin <wr_en_off,rd_en_on,wr_ack_off>...	193	1	100.00...	
bin <wr_en_on,rd_en_on,wr_ack_off>	178	1	100.00...	
bin <wr_en_on,rd_en_off,wr_ack_on>	553	1	100.00...	
bin <wr_en_on,rd_en_on,wr_ack_on>	268	1	100.00...	
illegal_bin wr_en_ack_on	0	-	-	
CROSS fifo_cg::wr_rd_overflow_cp	100.00%	100	100.00...	
bin <wr_en_off,rd_en_off,overflow_of...	410	1	100.00...	
bin <wr_en_on,rd_en_off,overflow_off...	573	1	100.00...	
bin <wr_en_off,rd_en_on,overflow_off...	193	1	100.00...	
bin <wr_en_on,rd_en_on,overflow_off...	272	1	100.00...	
bin <wr_en_on,rd_en_off,overflow_on...	377	1	100.00...	
bin <wr_en_on,rd_en_on,overflow_on...	174	1	100.00...	
illegal_bin wr_en_overflow_on	0	-	-	
CROSS fifo_cg::wr_rd_full_cp	100.00%	100	100.00...	
bin <wr_en_off,rd_en_off,full_off>	236	1	100.00...	
bin <wr_en_off,rd_en_off,full_on>	174	1	100.00...	
bin <wr_en_on,rd_en_off,full_off>	309	1	100.00...	
bin <wr_en_on,rd_en_off,full_on>	641	1	100.00...	
bin <wr_en_off,rd_en_on,full_off>	193	1	100.00...	
bin <wr_en_on,rd_en_on,full_off>	446	1	100.00...	
illegal_bin wr_en_overflow_on	0	-	-	



CROSS fifo_cg::wr_rd_empty_cp	100.00%	100	100.00...	
bin <wr_en_off,rd_en_off,empty_off>	392	1	100.00...	
bin <wr_en_on,rd_en_off,empty_off>	930	1	100.00...	
bin <wr_en_off,rd_en_on,empty_off>	183	1	100.00...	
bin <wr_en_on,rd_en_on,empty_off>	442	1	100.00...	
bin <wr_en_off,rd_en_off,empty_on>	18	1	100.00...	
bin <wr_en_on,rd_en_off,empty_on>	20	1	100.00...	
bin <wr_en_off,rd_en_on,empty_on>	10	1	100.00...	
bin <wr_en_on,rd_en_on,empty_on>	4	1	100.00...	
CROSS fifo_cg::wr_rd_almostfull_cp	100.00%	100	100.00...	
bin <wr_en_off,rd_en_off,almostfull_o...	318	1	100.00...	
bin <wr_en_on,rd_en_off,almostfull_of...	878	1	100.00...	
bin <wr_en_off,rd_en_on,almostfull_of...	116	1	100.00...	
bin <wr_en_on,rd_en_on,almostfull_of...	156	1	100.00...	
bin <wr_en_off,rd_en_off,almostfull_o...	92	1	100.00...	
bin <wr_en_on,rd_en_off,almostfull_o...	72	1	100.00...	
bin <wr_en_off,rd_en_on,almostfull_o...	77	1	100.00...	
bin <wr_en_on,rd_en_on,almostfull_on...	290	1	100.00...	
CROSS fifo_cg::wr_rd_almostempty_cp	100.00%	100	100.00...	
bin <wr_en_off,rd_en_off,almostempt...	398	1	100.00...	
bin <wr_en_on,rd_en_off,almostempty...	927	1	100.00...	
bin <wr_en_off,rd_en_on,almostempty...	179	1	100.00...	
bin <wr_en_on,rd_en_on,almostempty...	419	1	100.00...	
bin <wr_en_off,rd_en_off,almostempt...	12	1	100.00...	
bin <wr_en_on,rd_en_off,almostempty...	23	1	100.00...	
bin <wr_en_off,rd_en_on,almostempty...	14	1	100.00...	
bin <wr_en_on,rd_en_on,almostempty...	27	1	100.00...	



Code coverage:

```
Branch Coverage:
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Branches                        27        27         0   100.00%

Condition Coverage:
    Enabled Coverage              Bins   Covered    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Conditions                      24        24         0   100.00%
```

```
Statement Coverage:
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Statements                      31        31         0   100.00%


  Toggle Coverage:
      Enabled Coverage              Bins      Hits    Misses  Coverage
      ----------------              ----      ----    ------  --------
      Toggles                         20        20         0   100.00%

  ================================Toggle Details================================

  Toggle Coverage for instance /fifo_top/DUT --
                                                                  Node      1H->0L      0L->1H  "Coverage"
                                                                  ----------------------------------------
                                                             count[3-0]           1           1      100.00
                                                            rd_ptr[2-0]           1           1      100.00
                                                            wr_ptr[2-0]           1           1      100.00

  Total Node Count     =          10
  Toggled Node Count   =          10
  Untoggled Node Count =           0

  Toggle Coverage      =     100.00% (20 of 20 bins)
```

```
================================================================================
=== Instance: /fifo_top/fifoif
=== Design Unit: work.fifo_if
================================================================================
Toggle Coverage:
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Toggles                         86        86         0   100.00%

================================Toggle Details================================

Toggle Coverage for instance /fifo_top/fifoif --
                                                    Node      1H->0L      0L->1H  "Coverage"
                                                    ----------------------------------------
                                               almostempty           1           1      100.00
                                                almostfull           1           1      100.00
                                                       clk           1           1      100.00
                                              data_in[15-0]           1           1      100.00
                                             data_out[15-0]           1           1      100.00
                                                     empty           1           1      100.00
                                                      full           1           1      100.00
                                                  overflow           1           1      100.00
                                                     rd_en           1           1      100.00
                                                     rst_n           1           1      100.00
                                                 underflow           1           1      100.00
                                                    wr_ack           1           1      100.00
                                                     wr_en           1           1      100.00

Total Node Count     =         43
Toggled Node Count   =         43
Untoggled Node Count =          0

Toggle Coverage      =     100.00% (86 of 86 bins)
```