PROJECT 1

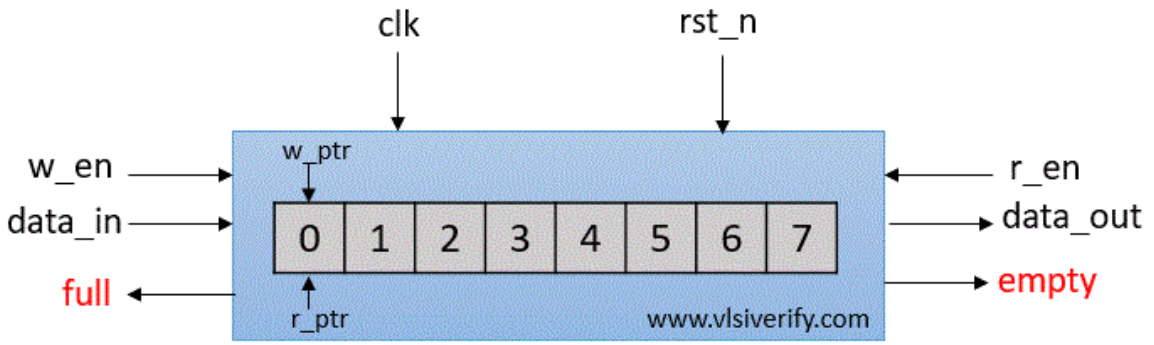
Synchronous FIFO Verification

Prepared by: Farah Haitham Saddik
Supervised by: Eng. Kareem Waseem

INTRODUCTION

A **First-In, First-Out (FIFO)** buffer is a fundamental element in digital system design used for data stream management between subsystems operating at different clock domains or data rates. It ensures sequential data integrity, following the principle that the first data word written into the buffer is the first one read out. FIFOs are widely utilized in communication interfaces**,** processor pipelines**,** and hardware accelerators, where synchronization and throughput consistency are essential.

In this project, a synchronous FIFO was designed in Verilog to support configurable data width and depth. The design integrates write/read control logic, pointers wraparound, and status flag generation for real-time monitoring of buffer occupancy. To validate the design's functionality, a SystemVerilog-based verification environment was developed, featuring constrained-random stimulus generation**,** assertion-based verification (SVA)**,** and a scoreboard-driven self-checking mechanism. Functional coverage was employed to ensure all operational and corner-case scenarios were exercised.



Synchronous FIFO

INTERFACE

FIFO_if.sv

```
interface FIFO_if(clk);
parameter FIFO_WIDTH = 16;
parameter FIFO_DEPTH = 8;
input clk;
logic [FIFO_WIDTH-1:0] data_in;
logic rst_n, wr_en, rd_en;
logic [FIFO_WIDTH-1:0] data_out;
logic wr_ack, overflow;
logic full, empty, almostfull, almostempty, underflow;
event input_driven;

modport DUT(input data_in, rst_n, wr_en, rd_en, clk,
            output data_out, wr_ack, overflow, full, empty, almostfull, almostempty, underflow);
modport TEST(output data_in, rst_n, wr_en, rd_en, input clk, input_driven,
             data_out, wr_ack, overflow, full, empty, almostfull, almostempty, underflow);
modport MONITOR(input data_in, rst_n, wr_en, rd_en, clk, input_driven ,
                data_out, wr_ack, overflow, full, empty, almostfull, almostempty, underflow);

endinterface
```

RTL CODE BEFORE MODIFICATIONS

```
module FIFO(data_in, wr_en, rd_en, clk, rst_n, full, empty, almostfull, almostempty, wr_ack, overflow, underflow, data_out);
parameter FIFO_WIDTH = 16;
parameter FIFO_DEPTH = 8;
input [FIFO_WIDTH-1:0] data_in;
input clk, rst_n, wr_en, rd_en;
output reg [FIFO_WIDTH-1:0] data_out;
output reg wr_ack, overflow;
output full, empty, almostfull, almostempty, underflow;

localparam max_fifo_addr = $clog2(FIFO_DEPTH);

reg [FIFO_WIDTH-1:0] mem [FIFO_DEPTH-1:0];

reg [max_fifo_addr-1:0] wr_ptr, rd_ptr;
reg [max_fifo_addr:0] count;

always @(posedge clk or negedge rst_n) begin
    if (!rst_n) begin
        wr_ptr <= 0;
        // NOT RESETING overflow FLAG & wr_ack COUNTER
    end
    else if (wr_en && count < FIFO_DEPTH) begin
        mem[wr_ptr] <= data_in;
        wr_ack <= 1;
        wr_ptr <= wr_ptr + 1;
    end
    else begin
        wr_ack <= 0;
        if (full & wr_en)
            overflow <= 1;
        else
            overflow <= 0;
    end
end
```

```
always @(posedge clk or negedge rst_n) begin
    if (!rst_n) begin
        rd_ptr <= 0;
        // NOT RESETING underflow FLAG
    end
    else if (rd_en && count != 0) begin
        data_out <= mem[rd_ptr];
        rd_ptr <= rd_ptr + 1;
    end
end

always @(posedge clk or negedge rst_n) begin
    if (!rst_n) begin
        count <= 0;
    end
    else begin
        // NOT CONSIDERING CONDITIONS WHERE ({wr_en, rd_en} == 2'b11)
        if  ( ({wr_en, rd_en} == 2'b10) && !full)
            count <= count + 1;
        else if ( ({wr_en, rd_en} == 2'b01) && !empty)
            count <= count - 1;
    end
end

assign full = (count == FIFO_DEPTH)? 1 : 0;
assign empty = (count == 0)? 1 : 0;
assign underflow = (empty && rd_en)? 1 : 0; // underflow IS A SEQUENTIAL FLAG
assign almostfull = (count == FIFO_DEPTH-2)? 1 : 0;
assign almostempty = (count == 1)? 1 : 0;

endmodule
```

RTL CODE AFTER MODIFICATIONS

```
module FIFO(FIFO_if.DUT fifo_if);

localparam max_fifo_addr = $clog2(fifo_if.FIFO_DEPTH);

reg [fifo_if.FIFO_WIDTH-1:0] mem [fifo_if.FIFO_DEPTH-1:0];

reg [max_fifo_addr-1:0] wr_ptr, rd_ptr;
reg [max_fifo_addr:0] count;

always @(posedge fifo_if.clk or negedge fifo_if.rst_n) begin
    if (!fifo_if.rst_n) begin
        wr_ptr <= 0;
        fifo_if.overflow <= 0;
        fifo_if.wr_ack <= 0;
    end
    else if (fifo_if.wr_en && count < fifo_if.FIFO_DEPTH) begin
        mem[wr_ptr] <= fifo_if.data_in;
        fifo_if.wr_ack <= 1;
        wr_ptr <= wr_ptr + 1;
        fifo_if.overflow <= 0;
    end
    else begin
        fifo_if.wr_ack <= 0;
        if (fifo_if.full & fifo_if.wr_en)
            fifo_if.overflow <= 1;
        else
            fifo_if.overflow <= 0;
    end
end

always @(posedge fifo_if.clk or negedge fifo_if.rst_n) begin
    if (!fifo_if.rst_n) begin
        rd_ptr <= 0;
        fifo_if.underflow <= 0;
    end
    else if (fifo_if.rd_en && count != 0) begin
        fifo_if.data_out <= mem[rd_ptr];
        rd_ptr <= rd_ptr + 1;
```

```
        fifo_if.underflow <= 0;
    end
    else begin
        if (fifo_if.empty && fifo_if.rd_en)
            fifo_if.underflow <= 1;
        else
            fifo_if.underflow <= 0;
    end
end

always @(posedge fifo_if.clk or negedge fifo_if.rst_n) begin
    if (!fifo_if.rst_n) begin
        count <= 0;
    end
    else begin
        if  ( ({fifo_if.wr_en, fifo_if.rd_en} == 2'b10) && !fifo_if.full)
            count <= count + 1;
        else if ( ({fifo_if.wr_en, fifo_if.rd_en} == 2'b01) && !fifo_if.empty)
            count <= count - 1;
        else if ( ({fifo_if.wr_en, fifo_if.rd_en} == 2'b11) && fifo_if.full)
            count <= count - 1;
        else if ( ({fifo_if.wr_en, fifo_if.rd_en} == 2'b11) && fifo_if.empty)
            count <= count + 1;
    end
end

assign fifo_if.full = (count == fifo_if.FIFO_DEPTH)? 1 : 0;
assign fifo_if.empty = (count == 0)? 1 : 0;
assign fifo_if.almostfull = (count == fifo_if.FIFO_DEPTH-1)? 1 : 0;
assign fifo_if.almostempty = (count == 1)? 1 : 0;
```

ASSERTIONS

```
`ifdef SIM

always_comb begin
    if(!fifo_if.rst_n) begin
        count_a: assert final(count == {(max_fifo_addr+1){1'b0}});
        wr_ptr_a: assert final(wr_ptr == {max_fifo_addr{1'b0}});
        rd_ptr_a: assert final(rd_ptr == {max_fifo_addr{1'b0}});
        overflow_a: assert final(fifo_if.overflow == 1'b0);
        wr_ack_a: assert final(fifo_if.wr_ack == 1'b0);
        underflow_a: assert final(fifo_if.underflow == 1'b0);
        full_a: assert final(fifo_if.full == 1'b0);
        empty_a: assert final(fifo_if.empty == 1'b1);
        almostfull_a: assert final(fifo_if.almostfull == 1'b0);
        almostempty_a: assert final(fifo_if.almostempty == 1'b0);
    end

    if(count == fifo_if.FIFO_DEPTH)
        full_aa: assert final(fifo_if.full == 1'b1);

    if(count == fifo_if.FIFO_DEPTH-1)
        almostfull_aa: assert final(fifo_if.almostfull == 1'b1);

    if(count == 1'b0)
        empty_aa: assert final(fifo_if.empty == 1'b1);

    if(count == 1'b1)
        almostempty_aa: assert final(fifo_if.almostempty == 1'b1);
end

property wr_ack_p;
    @(posedge fifo_if.clk) disable iff (!fifo_if.rst_n)
    (!fifo_if.full && fifo_if.wr_en) |=> fifo_if.wr_ack;
endproperty

property overflow_p;
    @(posedge fifo_if.clk) disable iff (!fifo_if.rst_n)
    (fifo_if.full && fifo_if.wr_en) |=> fifo_if.overflow;
endproperty
```

```
property underflow_p;
    @(posedge fifo_if.clk) disable iff (!fifo_if.rst_n)
    (fifo_if.empty && fifo_if.rd_en) |=> fifo_if.underflow;
endproperty

property count_rw_empty_p;
    @(posedge fifo_if.clk) disable iff (!fifo_if.rst_n)
    (fifo_if.empty && fifo_if.rd_en && fifo_if.wr_en) |=> (count == $past(count) + 1'b1);
endproperty

property count_rw_full_p;
    @(posedge fifo_if.clk) disable iff (!fifo_if.rst_n)
    (fifo_if.full && fifo_if.rd_en && fifo_if.wr_en) |=> (count == $past(count) - 1'b1);
endproperty

property count_w_p;
    @(posedge fifo_if.clk) disable iff (!fifo_if.rst_n)
    (!fifo_if.full && !fifo_if.rd_en && fifo_if.wr_en) |=> (count == $past(count) + 1'b1);
endproperty

property count_r_p;
    @(posedge fifo_if.clk) disable iff (!fifo_if.rst_n)
    (!fifo_if.empty && fifo_if.rd_en && !fifo_if.wr_en) |=> (count == $past(count) - 1'b1);
endproperty

property wr_ptr_p;
    @(posedge fifo_if.clk) disable iff (!fifo_if.rst_n)
    (!fifo_if.full && fifo_if.wr_en) |=> (wr_ptr == $past(wr_ptr) + 1'b1);
endproperty

property rd_ptr_p;
    @(posedge fifo_if.clk) disable iff (!fifo_if.rst_n)
    (!fifo_if.empty && fifo_if.rd_en) |=> (rd_ptr == $past(rd_ptr) + 1'b1);
endproperty
```

```
    wr_ack_pa: assert property(wr_ack_p);
    overflow_pa: assert property(overflow_p);
    underflow_pa: assert property(underflow_p);
    count_rw_empty_pa: assert property(count_rw_empty_p);
    count_rw_full_pa: assert property(count_rw_full_p);
    count_w_pa: assert property(count_w_p);
    count_r_pa: assert property(count_r_p);
    wr_ptr_pa: assert property(wr_ptr_p);
    rd_ptr_pa: assert property(rd_ptr_p);

    wr_ack_pc: cover property(wr_ack_p);
    overflow_pc: cover property(overflow_p);
    underflow_pc: cover property(underflow_p);
    count_rw_empty_pc: cover property(count_rw_empty_p);
    count_rw_full_pc: cover property(count_rw_full_p);
    count_w_pc: cover property(count_w_p);
    count_r_pc: cover property(count_r_p);
    wr_ptr_pc: cover property(wr_ptr_p);
    rd_ptr_pc: cover property(rd_ptr_p);

    `endif

    endmodule
```

FIFO_TRANSACTION CLASS

FIFO_transaction_pkg.sv

```
1   package FIFO_transaction_pkg;
2
3   class FIFO_transaction;
4       parameter FIFO_WIDTH = 16;
5       parameter FIFO_DEPTH = 8;
6       rand logic [FIFO_WIDTH-1:0] data_in;
7       rand logic rst_n, wr_en, rd_en;
8       logic [FIFO_WIDTH-1:0] data_out;
9       logic wr_ack, overflow, full, empty, almostfull, almostempty, underflow;
10      int RD_EN_ON_DIST;
11      int WR_EN_ON_DIST;
12
13      function new(int rd_dist = 30, wr_dist = 70);
14          RD_EN_ON_DIST = rd_dist;
15          WR_EN_ON_DIST = wr_dist;
16      endfunction
17
18      // FIFO_1
19      constraint rst_n_const{
20          rst_n dist {1 :/ 95, 0 :/ 5};
21      }
22
23      // FIFO_2
24      constraint wr_en_const{
25          wr_en dist {1 :/ WR_EN_ON_DIST, 0:/ (100 - WR_EN_ON_DIST)};
26      }
27
28      // FIFO_3
29      constraint rd_en_const{
30          rd_en dist {1 :/ RD_EN_ON_DIST, 0:/ (100 - RD_EN_ON_DIST)};
31      }
32
33  endclass
34
35  endpackage
```

SHARED PACKEGE

shared_pkg.sv

```
1   package shared_pkg;
2       int error_count;
3       int all_error_count;
4       int correct_count;
5       int all_correct_count;
6       bit test_finished;
7       event input_driven;
8   endpackage
```

FIFO_SCOREBOARD CLASS

FIFO_scoreboard.sv

```sv
package FIFO_scoreboard_pkg;
import FIFO_transaction_pkg::*;
import shared_pkg::*;

class FIFO_scoreboard;
    parameter FIFO_WIDTH = 16;
    logic [FIFO_WIDTH-1:0] data_out_ref;
    logic wr_ack_ref, overflow_ref, full_ref, empty_ref, almostfull_ref, almostempty_ref, underflow_ref;
    logic[FIFO_WIDTH-1:0] qmem [$];

    function void check_data(FIFO_transaction ftr);
        reference_model(ftr);

        if (ftr.data_out != data_out_ref) begin
            $display("ERROR!! - data_out_expected = %h , data_out = %h", data_out_ref, ftr.data_out);
            error_count++;
            all_error_count++;
        end
        else begin
            correct_count++;
            all_correct_count++;
        end

        if (ftr.wr_ack != wr_ack_ref) begin
            $display("ERROR!! - wr_ack_expected = %h , wr_ack = %h", wr_ack_ref, ftr.wr_ack);
            all_error_count++;
        end
        else all_correct_count++;

        if (ftr.overflow != overflow_ref) begin
            $display("ERROR!! - overflow_expected = %h , overflow = %h", overflow_ref, ftr.overflow);
            all_error_count++;
        end
        else all_correct_count++;
```

```sv
        if (ftr.full != full_ref) begin
            $display("ERROR!! - full_expected = %h , full = %h", full_ref, ftr.full);
            all_error_count++;
        end
        else all_correct_count++;

        if (ftr.empty != empty_ref) begin
            $display("ERROR!! - empty_expected = %h , empty = %h", empty_ref, ftr.empty);
            all_error_count++;
        end
        else all_correct_count++;

        if (ftr.almostfull != almostfull_ref) begin
            $display("ERROR!! - almostfull_expected = %h , almostfull = %h", almostfull_ref, ftr.almostfull);
            all_error_count++;
        end
        else all_correct_count++;

        if (ftr.almostempty != almostempty_ref) begin
            $display("ERROR!! - almostempty_expected = %h , almostempty = %h", almostempty_ref, ftr.almostempty);
            all_error_count++;
        end
        else all_correct_count++;

        if (ftr.underflow != underflow_ref) begin
            $display("ERROR!! - underflow_expected = %h , underflow = %h", underflow_ref, ftr.underflow);
            all_error_count++;
        end
        else all_correct_count++;

    endfunction
```

```
68      function void reference_model(FIFO_transaction ftr);
69          if (!ftr.rst_n) begin
70              wr_ack_ref = 0;
71              overflow_ref = 0;
72              full_ref = 0;
73              empty_ref = 1;
74              almostfull_ref = 0;
75              almostempty_ref = 0;
76              underflow_ref = 0;
77              qmem.delete();
78          end
79          else begin
80              wr_ack_ref = (ftr.wr_en && qmem.size() < ftr.FIFO_DEPTH);
81              overflow_ref = (ftr.wr_en && qmem.size() == ftr.FIFO_DEPTH);
82              underflow_ref = (ftr.rd_en && qmem.size() == 0);
83
84              if (ftr.rd_en && !empty_ref) data_out_ref = qmem.pop_front();
85              if (ftr.wr_en && !full_ref) qmem.push_back(ftr.data_in);
86
87              full_ref = (qmem.size() == ftr.FIFO_DEPTH);
88              empty_ref = (qmem.size() == 0);
89              almostfull_ref = (qmem.size() == ftr.FIFO_DEPTH - 1);
90              almostempty_ref = (qmem.size() == 1);
91
92          end
93      endfunction
94
95  endclass
96
97  endpackage
```

FIFO_COVERAGE CLASS

```
1   package FIFO_coverage_pkg;
2   import FIFO_transaction_pkg::*;
3
4   class FIFO_coverage;
5       FIFO_transaction F_cvg_txn;
6
7       covergroup Cov;
8           wr_en_cp: coverpoint F_cvg_txn.wr_en;
9           rd_en_cp: coverpoint F_cvg_txn.rd_en;
10          wr_ack_cp: coverpoint F_cvg_txn.wr_ack;
11          overflow_cp: coverpoint F_cvg_txn.overflow;
12          full_cp: coverpoint F_cvg_txn.full;
13          empty_cp: coverpoint F_cvg_txn.empty;
14          almostfull_cp: coverpoint F_cvg_txn.almostfull;
15          almostempty_cp: coverpoint F_cvg_txn.almostempty;
16          underflow_cp: coverpoint F_cvg_txn.underflow;
17
18          wr_ack_cr: cross wr_en_cp, rd_en_cp, wr_ack_cp{
19              illegal_bins wr_ack_illegal = binsof(wr_en_cp) intersect {0} && binsof(wr_ack_cp) intersect {1};
20          }
21          overflow_cr: cross wr_en_cp, rd_en_cp, overflow_cp{
22              illegal_bins overflow_illegal = binsof(wr_en_cp) intersect {0} && binsof(overflow_cp) intersect {1};
23          }
24          full_cr: cross wr_en_cp, rd_en_cp, full_cp{
25              illegal_bins full_illegal = binsof(rd_en_cp) intersect {1} && binsof(full_cp) intersect {1};
26          }
27          empty_cr: cross wr_en_cp, rd_en_cp, empty_cp;
28          almostfull_cr: cross wr_en_cp, rd_en_cp, almostfull_cp;
29          almostempty_cr: cross wr_en_cp, rd_en_cp, almostempty_cp;
30          underflow_cr: cross wr_en_cp, rd_en_cp, underflow_cp{
31              illegal_bins underflow_illegal = binsof(rd_en_cp) intersect {0} && binsof(underflow_cp) intersect {1};
32          }
33
```

```
36      function new();
37          Cov = new();
38      endfunction
39
40      function void sample_data(FIFO_transaction F_txn);
41          F_cvg_txn = F_txn;
42          Cov.sample();
43      endfunction
44  endclass
45
46  endpackage
```

FIFO_MONITOR CLASS

```
1   import shared_pkg::*;
2   import FIFO_scoreboard_pkg::*;
3   import FIFO_transaction_pkg::*;
4   import FIFO_coverage_pkg::*;
5
6   module FIFO_monitor (FIFO_if.MONITOR fifo_if);
7   FIFO_coverage fc = new();
8   FIFO_transaction ft = new();
9   FIFO_scoreboard fs = new();
10
11  initial begin
12      forever begin
13          @(input_driven);
14          @(negedge fifo_if.clk);
15          ft.data_in = fifo_if.data_in;
16          ft.rst_n = fifo_if.rst_n;
17          ft.wr_en = fifo_if.wr_en;
18          ft.rd_en = fifo_if.rd_en;
19          ft.data_out = fifo_if.data_out;
20          ft.wr_ack = fifo_if.wr_ack;
21          ft.overflow = fifo_if.overflow;
22          ft.full = fifo_if.full;
23          ft.empty = fifo_if.empty;
24          ft.almostfull = fifo_if.almostfull;
25          ft.almostempty = fifo_if.almostempty;
26          ft.underflow = fifo_if.underflow;
27
```

```
        fork
            begin
                fc.sample_data(ft);
            end

            begin
                fs.check_data(ft);
            end
        join

        if (test_finished) begin
            $display(" TEST FINISHED: correct_count = %d, error_count = %d", correct_count, error_count);
            $display(" TEST FINISHED: all_correct_count = %d, all_error_count = %d", all_correct_count, all_error_count);
            $stop;
        end
    end
end

endmodule
```

TESTBENCH

FIFO_tb.sv

```
import FIFO_transaction_pkg::*;
import shared_pkg::*;

module FIFO_tb (FIFO_if.TEST fifo_if);

FIFO_transaction obj = new();

initial begin
    assert_reset();
    -> input_driven;

    repeat(10000) begin
        assert(obj.randomize());
        fifo_if.rst_n = obj.rst_n;
        fifo_if.data_in = obj.data_in;
        fifo_if.wr_en = obj.wr_en;
        fifo_if.rd_en = obj.rd_en;
        @(negedge fifo_if.clk);
        -> input_driven;
    end
    test_finished = 1;
end

task assert_reset();
    fifo_if.rst_n = 0;
    fifo_if.data_in = 0;
    fifo_if.wr_en = 0;
    fifo_if.rd_en = 0;
    @(negedge fifo_if.clk);
    fifo_if.rst_n = 1;
endtask

endmodule
```

TOP MODULE

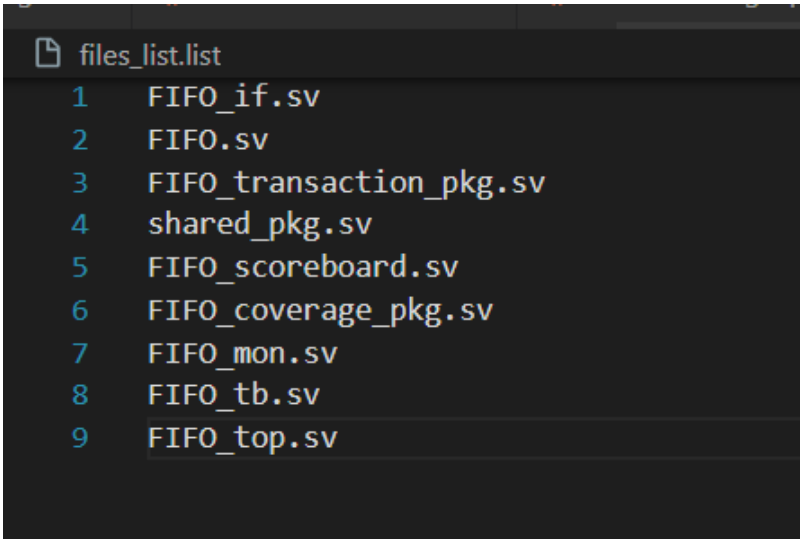
FIFO_top.sv

```
1   module FIFO_top();
2   bit clk;
3   initial begin
4       forever #1 clk = ~clk;
5   end
6
7   FIFO_if fifo_if(clk);
8
9   FIFO dut(fifo_if);
10  FIFO_tb tb(fifo_if);
11  FIFO_monitor mon(fifo_if);
12
13  endmodule
```

DO FILE

run.do

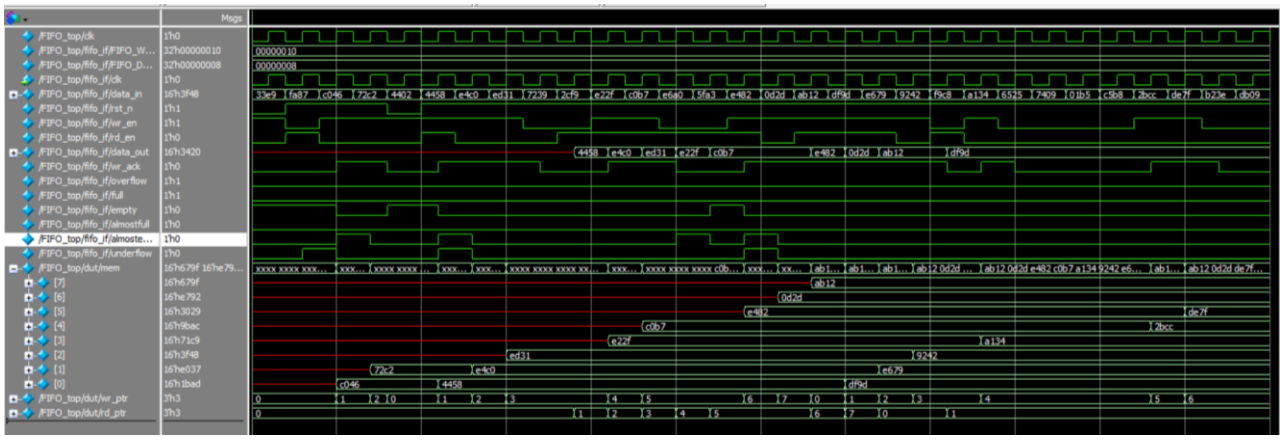
```
1   vlib work
2   vlog +define+SIM -f files_list.list +cover -covercells
3   vsim -voptargs=+acc work.FIFO_top -cover
4   add wave *
5   run 0
6   add wave -position insertpoint  \
7   sim:/FIFO_top/fifo_if/FIFO_WIDTH \
8   sim:/FIFO_top/fifo_if/FIFO_DEPTH \
9   sim:/FIFO_top/fifo_if/clk \
10  sim:/FIFO_top/fifo_if/data_in \
11  sim:/FIFO_top/fifo_if/rst_n \
12  sim:/FIFO_top/fifo_if/wr_en \
13  sim:/FIFO_top/fifo_if/rd_en \
14  sim:/FIFO_top/fifo_if/data_out \
15  sim:/FIFO_top/fifo_if/wr_ack \
16  sim:/FIFO_top/fifo_if/overflow \
17  sim:/FIFO_top/fifo_if/full \
18  sim:/FIFO_top/fifo_if/empty \
19  sim:/FIFO_top/fifo_if/almostfull \
20  sim:/FIFO_top/fifo_if/almostempty \
21  sim:/FIFO_top/fifo_if/underflow
22  add wave -position insertpoint  \
23  sim:/FIFO_top/dut/mem \
24  sim:/FIFO_top/dut/wr_ptr \
25  sim:/FIFO_top/dut/rd_ptr
26  run -all
27  coverage save FIFO_top.ucdb -onexit
```

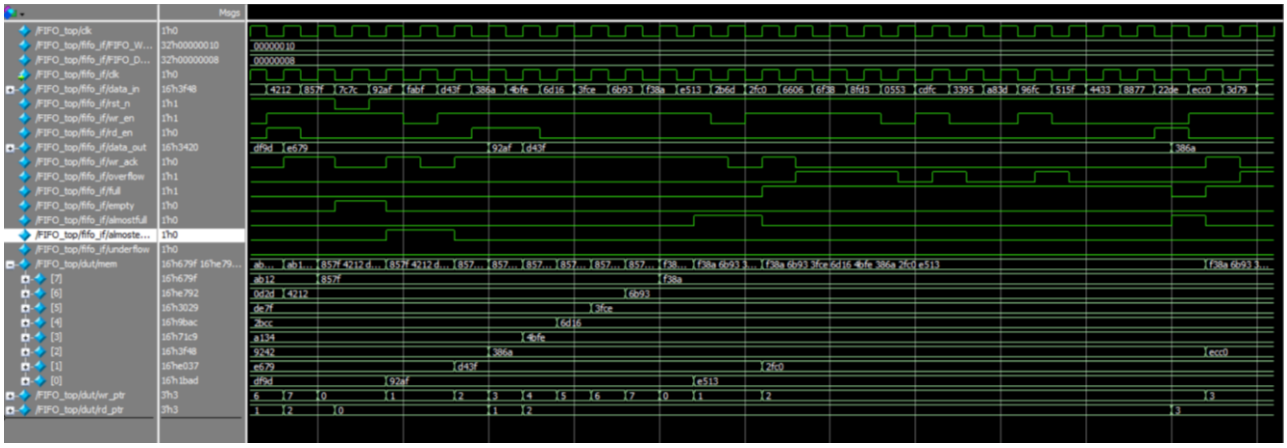


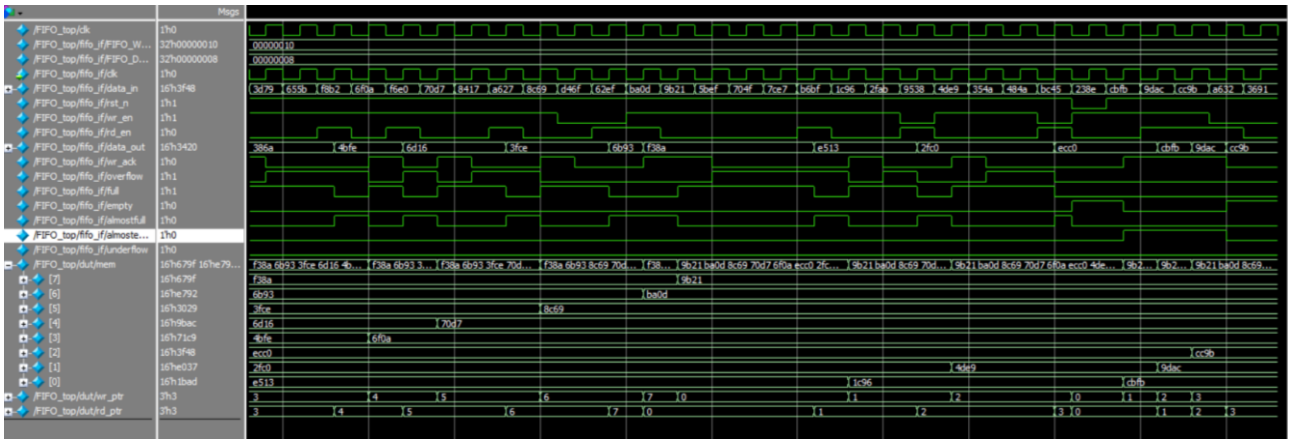
VERIFICATION PLAN

Label	Design Requirement Description	Stimulus Generation	Functional Coverage	Functionality Check
FIFO_1	When the reset is asserted, the flags (ful -almostfull - almostempty - overflow - underflow) should be low and empty should be high	Directed at the start of the simulation, and then randomized with constraint that drives the reset to be off 95% of the simulation time	-	Immediate assertion & A checker in the reference model to check for the asynch reset functionality
FIFO_2	When the reset is deasserted and wr_en is asserted, the input data_in should be inserted in the fifo where the pointer wr_ptr is pointing in & the output wr_ack should be high if the fifo is not full	Randomizition for data_in ,randomized with constraint that drives the wr_en to be on WR_EN_ON_DIST% of the simulation time	Covers all values of wr_en & wr_ack	Concurrent assertion & A checker in the reference model to check for the wr_ack
FIFO_3	When the reset is deasserted and rd_en is asserted, the output data_out should take the value of where the pointer rd_ptr is pointing in	Randomizition for rd_en with constraint that drives the it to be on RD_EN_ON_DIST% of the simulation time	Covers all values of rd_en	A checker in the reference model to make sure the output data_out is correct
FIFO_4	When the reset is deasserted and the fifo is full, the ouput full should be high	-	Covers all values of full	Immediate assertion & A checker in the reference model to check for the full flag
FIFO_5	When the reset is deasserted and the fifo is almost full (only one place left to write in), the ouput almostfull should be high	-	Covers all values of almostfull	Immediate assertion & A checker in the reference model to check for the almostflag flag
FIFO_6	When the reset is deasserted and the fifo is empty, the ouput empty should be high	-	Covers all values of empty	Immediate assertion & A checker in the reference model to check for the empty flag
FIFO_7	When the reset is deasserted and the fifo is almost empty(only one place left to be read), the ouput almostempty should be high	-	Covers all values of almostempty	Immediate assertion & A checker in the reference model to check for the almostempty flag
FIFO_8	When the reset is deasserted and the fifo is full and wr_en is high, the ouput overflow should be high	-	Covers all values of overflow	Concurrent assertion & A checker in the reference model to check for the overflow flag
FIFO_9	When the reset is deasserted and the fifo is empty and rd_en is high, the ouput underflow should be high	-	Covers all values of underflow	Concurrent assertion & A checker in the reference model to check for the underflow flag
FIFO_10	When the reset is deasserted and the wr_en is high when the fifo is not full, the count & wr_ptr should increment by	-	-	Concurrent assertion to check for count & wr_ptr
FIFO_11	When the reset is deasserted and the rd_en is high when the fifo is not empty, the count should decrement by one & wr_ptr should increment by one	-	-	Concurrent assertion to check for count & rd_ptr

WAVEFORM







COVERAGE REPORT

```
# ================================================================================
# === Instance: /FIFO_top/fifo_if
# === 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/fifo_if --
#
#                                             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)
#
# ----------------------------------------------------------------------------------
```

```
# Assertion Coverage:
#     Assertions                      23        23         0   100.00%
# ----------------------------------------------------------------------
# Name                 File(Line)                    Failure      Pass
#                                                    Count        Count
# ----------------------------------------------------------------------
# /FIFO_top/dut/count_a
#                      FIFO.sv(81)                         0          1
# /FIFO_top/dut/wr_ptr_a
#                      FIFO.sv(82)                         0          1
# /FIFO_top/dut/rd_ptr_a
#                      FIFO.sv(83)                         0          1
# /FIFO_top/dut/overflow_a
#                      FIFO.sv(84)                         0          1
# /FIFO_top/dut/wr_ack_a
#                      FIFO.sv(85)                         0          1
# /FIFO_top/dut/underflow_a
#                      FIFO.sv(86)                         0          1
# /FIFO_top/dut/full_a FIFO.sv(87)                         0          1
# /FIFO_top/dut/empty_a
#                      FIFO.sv(88)                         0          1
# /FIFO_top/dut/almostfull_a
#                      FIFO.sv(89)                         0          1
# /FIFO_top/dut/almostempty_a
#                      FIFO.sv(90)                         0          1
# /FIFO_top/dut/full_aa
#                      FIFO.sv(94)                         0          1
# /FIFO_top/dut/almostfull_aa
#                      FIFO.sv(97)                         0          1
# /FIFO_top/dut/empty_aa
#                      FIFO.sv(100)                        0          1
# /FIFO_top/dut/almostempty_aa
#                      FIFO.sv(103)                        0          1
# /FIFO_top/dut/wr_ack_pa
#                      FIFO.sv(151)                        0          1
# /FIFO_top/dut/overflow_pa
#                      FIFO.sv(152)                        0          1
# /FIFO_top/dut/underflow_pa
#                      FIFO.sv(153)                        0          1
# /FIFO_top/dut/count_rw_empty_pa
#                      FIFO.sv(154)                        0          1
# /FIFO_top/dut/count_rw_full_pa
#                      FIFO.sv(155)                        0          1
# /FIFO_top/dut/count_w_pa
#                      FIFO.sv(156)                        0          1
# /FIFO_top/dut/count_r_pa
#                      FIFO.sv(157)                        0          1
```

```
# /FIFO_top/dut/wr_ptr_pa
#                    FIFO.sv(158)                      0          1
# /FIFO_top/dut/rd_ptr_pa
#                    FIFO.sv(159)                      0          1
# Branch Coverage:
#     Enabled Coverage              Bins      Hits    Misses  Coverage
#     ----------------              ----      ----    ------  --------
#     Branches                        35        35         0   100.00%
#
# ================================Branch Details================================
#
# Branch Coverage for instance /FIFO_top/dut
#
#     Line         Item                     Count     Source
#     ----         ----                     -----     ------
#   File FIFO.sv
# ------------------------------------IF Branch------------------------------------
#     18                                   10466     Count coming in to IF
#     18             1                       947       if (!fifo_if.rst_n) begin
#
#     23             1                      5043       else if (fifo_if.wr_en && count < fifo_if.FIFO_DEPTH) begin
#
#     29             1                      4476       else begin
#
# Branch totals: 3 hits of 3 branches = 100.00%
#
# ------------------------------------IF Branch------------------------------------
#     31                                    4476     Count coming in to IF
#     31             1                      1564       if (fifo_if.full & fifo_if.wr_en)
#
#     33             1                      2912       else
#
# Branch totals: 2 hits of 2 branches = 100.00%
#
# ------------------------------------IF Branch------------------------------------
#     39                                   10466     Count coming in to IF
#     39             1                       947       if (!fifo_if.rst_n) begin
#
#     43             1                      2668       else if (fifo_if.rd_en && count != 0) begin
#
#     48             1                      6851       else begin
#
# Branch totals: 3 hits of 3 branches = 100.00%
#
# ------------------------------------IF Branch------------------------------------
#     49                                    6851     Count coming in to IF
#     49             1                       222       if (fifo_if.empty && fifo_if.rd_en)
#
#     51             1                      6629       else
#
# Branch totals: 2 hits of 2 branches = 100.00%
#
# ------------------------------------IF Branch------------------------------------
#     57                                    9274     Count coming in to IF
#     57             1                       942       if (!fifo_if.rst_n) begin
#
#     60             1                      8332       else begin
#
# Branch totals: 2 hits of 2 branches = 100.00%
#
# ------------------------------------IF Branch------------------------------------
#     61                                    8332     Count coming in to IF
#     61             1                      3514       if ( ({fifo_if.wr_en, fifo_if.rd_en} == 2'b10) && !fifo_if.full)
#
#     63             1                       816       else if ( ({fifo_if.wr_en, fifo_if.rd_en} == 2'b01) && !fifo_if.empty)
#
#     65             1                       477       else if ( ({fifo_if.wr_en, fifo_if.rd_en} == 2'b11) && fifo_if.full)
#
#     67             1                       154       else if ( ({fifo_if.wr_en, fifo_if.rd_en} == 2'b11) && fifo_if.empty)
#
#                                          3371     All False Count
# Branch totals: 5 hits of 5 branches = 100.00%
#
# ------------------------------------IF Branch------------------------------------
#     72                                    5407     Count coming in to IF
#     72             1                       818     assign fifo_if.full = (count == fifo_if.FIFO_DEPTH)? 1 : 0;
#
#     72             2                      4589     assign fifo_if.full = (count == fifo_if.FIFO_DEPTH)? 1 : 0;
#
# Branch totals: 2 hits of 2 branches = 100.00%
#
# ------------------------------------IF Branch------------------------------------
#     73                                    5407     Count coming in to IF
#     73             1                       522     assign fifo_if.empty = (count == 0)? 1 : 0;
#
#     73             2                      4885     assign fifo_if.empty = (count == 0)? 1 : 0;
#
# Branch totals: 2 hits of 2 branches = 100.00%
#
```

```
#
# ------------------------------------IF Branch------------------------------------
#     74                                    5407     Count coming in to IF
#     74              1                     1079     assign fifo_if.almostfull = (count == fifo_if.FIFO_DEPTH-1)? 1 : 0;
#
#     74              2                     4328     assign fifo_if.almostfull = (count == fifo_if.FIFO_DEPTH-1)? 1 : 0;
#
# Branch totals: 2 hits of 2 branches = 100.00%
#
# ------------------------------------IF Branch------------------------------------
#     75                                    5407     Count coming in to IF
#     75              1                      591     assign fifo_if.almostempty = (count == 1)? 1 : 0;
#
#     75              2                     4816     assign fifo_if.almostempty = (count == 1)? 1 : 0;
#
# Branch totals: 2 hits of 2 branches = 100.00%
#
# ------------------------------------IF Branch------------------------------------
#     80                                    9728     Count coming in to IF
#     80              1                      925      if(!fifo_if.rst_n) begin
#
#                                           8803     All False Count
# Branch totals: 2 hits of 2 branches = 100.00%
#
# ------------------------------------IF Branch------------------------------------
#     93                                    9728     Count coming in to IF
#     93              1                     1921      if(count == fifo_if.FIFO_DEPTH)
#
#                                           7807     All False Count
# Branch totals: 2 hits of 2 branches = 100.00%
#
# ------------------------------------IF Branch------------------------------------
#     96                                    9728     Count coming in to IF
#     96              1                     1728      if(count == fifo_if.FIFO_DEPTH-1)
#
#                                           8000     All False Count
# Branch totals: 2 hits of 2 branches = 100.00%
#
# ------------------------------------IF Branch------------------------------------
#     99                                    9728     Count coming in to IF
#     99              1                     1095      if(count == 1'b0)
#
#                                           8633     All False Count
# Branch totals: 2 hits of 2 branches = 100.00%
#
# ------------------------------------IF Branch------------------------------------
#     102                                   9728     Count coming in to IF
```

```
#     80                                    9728     Count coming in to IF
#     80              1                      925      if(!fifo_if.rst_n) begin
#
#                                           8803     All False Count
# Branch totals: 2 hits of 2 branches = 100.00%
#
# ------------------------------------IF Branch------------------------------------
#     93                                    9728     Count coming in to IF
#     93              1                     1921      if(count == fifo_if.FIFO_DEPTH)
#
#                                           7807     All False Count
# Branch totals: 2 hits of 2 branches = 100.00%
#
# ------------------------------------IF Branch------------------------------------
#     96                                    9728     Count coming in to IF
#     96              1                     1728      if(count == fifo_if.FIFO_DEPTH-1)
#
#                                           8000     All False Count
# Branch totals: 2 hits of 2 branches = 100.00%
#
# ------------------------------------IF Branch------------------------------------
#     99                                    9728     Count coming in to IF
#     99              1                     1095      if(count == 1'b0)
#
#                                           8633     All False Count
# Branch totals: 2 hits of 2 branches = 100.00%
#
# ------------------------------------IF Branch------------------------------------
#     102                                   9728     Count coming in to IF
#     102             1                     1001      if(count == 1'b1)
#
#                                           8727     All False Count
# Branch totals: 2 hits of 2 branches = 100.00%
#
```

```
# Statement Coverage:
#     Enabled Coverage              Bins      Hits    Misses  Coverage
#     ----------------              ----      ----    ------  --------
#     Statements                      30        30         0   100.00%
#
# ================================Statement Details================================
#
# Statement Coverage for instance /FIFO_top/dut --
#
#     Line         Item                      Count     Source
#     ----         ----                      -----     ------
#   File FIFO.sv
#     8                                                module FIFO(FIFO_if.DUT fifo_if);
#
#     9
#
#     10                                               localparam max_fifo_addr = $clog2(fifo_if.FIFO_DEPTH);
#
#     11
#
#     12                                               reg [fifo_if.FIFO_WIDTH-1:0] mem [fifo_if.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              1                     10466      always @(posedge fifo_if.clk or negedge fifo_if.rst_n) begin
#
#     18                                                if (!fifo_if.rst_n) begin
#
#     19              1                       947       wr_ptr <= 0;
#
#     20              1                       947       fifo_if.overflow <= 0;
#
#     21              1                       947       fifo_if.wr_ack <= 0;
#
#     22                                               end
#
#     23                                               else if (fifo_if.wr_en && count < fifo_if.FIFO_DEPTH) begin
#
#     24              1                      5043       mem[wr_ptr] <= fifo_if.data_in;
#
#     25              1                      5043       fifo_if.wr_ack <= 1;
#
#     26              1                      5043       wr_ptr <= wr_ptr + 1;
#
#     27              1                      5043       fifo_if.overflow <= 0;
#
#     28                                               end
#
#     29                                               else begin
#
#     30              1                      4476       fifo_if.wr_ack <= 0;
#
#     31                                                if (fifo_if.full & fifo_if.wr_en)
#
#     32              1                      1564        fifo_if.overflow <= 1;
#
#     33                                                else
#
#     34              1                      2912        fifo_if.overflow <= 0;
#
#     35                                               end
#
#     36                                               end
#
#     37
#
#     38              1                     10466      always @(posedge fifo_if.clk or negedge fifo_if.rst_n) begin
#
#     39                                                if (!fifo_if.rst_n) begin
#
#     40              1                       947       rd_ptr <= 0;
#
#     41              1                       947       fifo_if.underflow <= 0;
#
#     42                                               end
#
#     43                                               else if (fifo_if.rd_en && count != 0) begin
#
#     44              1                      2668       fifo_if.data_out <= mem[rd_ptr];
#
#     45              1                      2668       rd_ptr <= rd_ptr + 1;
#
#     46              1                      2668       fifo_if.underflow <= 0;
#
#     47                                               end
#
```

```
#
#     48                                        else begin
#
#     49                                         if (fifo_if.empty && fifo_if.rd_en)
#
#     50              1                   222     fifo_if.underflow <= 1;
#
#     51                                         else
#
#     52              1                  6629     fifo_if.underflow <= 0;
#
#     53                                        end
#
#     54                                       end
#
#     55
#
#     56              1                  9274    always @(posedge fifo_if.clk or negedge fifo_if.rst_n) begin
#
#     57                                         if (!fifo_if.rst_n) begin
#
#     58              1                   942     count <= 0;
#
#     59                                        end
#
#     60                                        else begin
#
#     61                                         if ( ({fifo_if.wr_en, fifo_if.rd_en} == 2'b10) && !fifo_if.full)
#
#     62              1                  3514     count <= count + 1;
#
#     63                                        else if ( ({fifo_if.wr_en, fifo_if.rd_en} == 2'b01) && !fifo_if.empty)
#
#     64              1                   816     count <= count - 1;
#
#     65                                        else if ( ({fifo_if.wr_en, fifo_if.rd_en} == 2'b11) && fifo_if.full)
#
#     66              1                   477     count <= count - 1;
#
#     67                                        else if ( ({fifo_if.wr_en, fifo_if.rd_en} == 2'b11) && fifo_if.empty)
#
#     68              1                   154     count <= count + 1;
#
#     69                                        end
#
#     70                                       end
```

```
#
#     57                                          if (!fifo_if.rst_n) begin
#
#     58              1                   942      count <= 0;
#
#     59                                          end
#
#     60                                          else begin
#
#     61                                           if ( ({fifo_if.wr_en, fifo_if.rd_en} == 2'b10) && !fifo_if.full)
#
#     62              1                  3514       count <= count + 1;
#
#     63                                          else if ( ({fifo_if.wr_en, fifo_if.rd_en} == 2'b01) && !fifo_if.empty)
#
#     64              1                   816       count <= count - 1;
#
#     65                                          else if ( ({fifo_if.wr_en, fifo_if.rd_en} == 2'b11) && fifo_if.full)
#
#     66              1                   477       count <= count - 1;
#
#     67                                          else if ( ({fifo_if.wr_en, fifo_if.rd_en} == 2'b11) && fifo_if.empty)
#
#     68              1                   154       count <= count + 1;
#
#     69                                          end
#
#     70                                         end
#
#     71
#
#     72              1                  5408    assign fifo_if.full = (count == fifo_if.FIFO_DEPTH)? 1 : 0;
#
#     73              1                  5408    assign fifo_if.empty = (count == 0)? 1 : 0;
#
#     74              1                  5408    assign fifo_if.almostfull = (count == fifo_if.FIFO_DEPTH-1)? 1 : 0;
#
#     75              1                  5408    assign fifo_if.almostempty = (count == 1)? 1 : 0;
#
#     76
#
#     77                                         `ifdef SIM
#
#     78
#
#     79              1                  9728    always_comb begin
#
```

```
#
# Covergroup Coverage:
#     Covergroups                    1        na        na   100.00%
#         Coverpoints/Crosses       16        na        na        na
#             Covergroup Bins       66        66         0   100.00%
# ----------------------------------------------------------------------------------------------------
# Covergroup                                          Metric       Goal       Bins    Status
#
# ----------------------------------------------------------------------------------------------------
#  TYPE /FIFO_coverage_pkg/FIFO_coverage/Cov        100.00%        100          -    Covered
#     covered/total bins:                              66         66          -
#     missing/total bins:                               0         66          -
#     % Hit:                                       100.00%        100          -
#     Coverpoint wr_en_cp                          100.00%        100          -    Covered
#         covered/total bins:                           2          2          -
#         missing/total bins:                           0          2          -
#         % Hit:                                   100.00%        100          -
#         bin auto[0]                                3051          1          -    Covered
#         bin auto[1]                                6950          1          -    Covered
#     Coverpoint rd_en_cp                          100.00%        100          -    Covered
#         covered/total bins:                           2          2          -
#         missing/total bins:                           0          2          -
#         % Hit:                                   100.00%        100          -
#         bin auto[0]                                6954          1          -    Covered
#         bin auto[1]                                3047          1          -    Covered
#     Coverpoint wr_ack_cp                         100.00%        100          -    Covered
#         covered/total bins:                           2          2          -
#         missing/total bins:                           0          2          -
#         % Hit:                                   100.00%        100          -
#         bin auto[0]                                4958          1          -    Covered
#         bin auto[1]                                5043          1          -    Covered
#     Coverpoint overflow_cp                       100.00%        100          -    Covered
#         covered/total bins:                           2          2          -
#         missing/total bins:                           0          2          -
#         % Hit:                                   100.00%        100          -
#         bin auto[0]                                8437          1          -    Covered
#         bin auto[1]                                1564          1          -    Covered
#     Coverpoint full_cp                           100.00%        100          -    Covered
#         covered/total bins:                           2          2          -
#         missing/total bins:                           0          2          -
#         % Hit:                                   100.00%        100          -
#         bin auto[0]                                7576          1          -    Covered
#         bin auto[1]                                2425          1          -    Covered
#     Coverpoint empty_cp                          100.00%        100          -    Covered
#         covered/total bins:                           2          2          -
#         missing/total bins:                           0          2          -
#         % Hit:                                   100.00%        100          -
#         bin auto[0]                                9206          1          -    Covered
#         bin auto[1]                                 795          1          -    Covered
#     Coverpoint almostfull_cp                     100.00%        100          -    Covered
#         covered/total bins:                           2          2          -
#         missing/total bins:                           0          2          -
#         % Hit:                                   100.00%        100          -
#         bin auto[0]                                8264          1          -    Covered
#         bin auto[1]                                1737          1          -    Covered
#     Coverpoint almostempty_cp                    100.00%        100          -    Covered
#         covered/total bins:                           2          2          -
#         missing/total bins:                           0          2          -
#         % Hit:                                   100.00%        100          -
#         bin auto[0]                                9001          1          -    Covered
#         bin auto[1]                                1000          1          -    Covered
#     Coverpoint underflow_cp                      100.00%        100          -    Covered
#         covered/total bins:                           2          2          -
#         missing/total bins:                           0          2          -
#         % Hit:                                   100.00%        100          -
#         bin auto[0]                                9779          1          -    Covered
#         bin auto[1]                                 222          1          -    Covered
#     Cross wr_ack_cr                              100.00%        100          -    Covered
#         covered/total bins:                           6          6          -
#         missing/total bins:                           0          6          -
#         % Hit:                                   100.00%        100          -
#         Auto, Default and User Defined Bins:
#             bin <auto[1],auto[1],auto[1]>          1529          1          -    Covered
#             bin <auto[1],auto[0],auto[1]>          3514          1          -    Covered
#             bin <auto[1],auto[1],auto[0]>           582          1          -    Covered
#             bin <auto[0],auto[1],auto[0]>           936          1          -    Covered
#             bin <auto[1],auto[0],auto[0]>          1325          1          -    Covered
#             bin <auto[0],auto[0],auto[0]>          2115          1          -    Covered
#         Illegal and Ignore Bins:
#             illegal_bin wr_ack_illegal                0                     -    ZERO
#     Cross overflow_cr                            100.00%        100          -    Covered
#         covered/total bins:                           6          6          -
#         missing/total bins:                           0          6          -
#         % Hit:                                   100.00%        100          -
#         Auto, Default and User Defined Bins:
#             bin <auto[1],auto[1],auto[1]>           477          1          -    Covered
#             bin <auto[1],auto[0],auto[1]>          1087          1          -    Covered
#             bin <auto[1],auto[1],auto[0]>          1634          1          -    Covered
#             bin <auto[0],auto[1],auto[0]>           936          1          -    Covered
#             bin <auto[1],auto[0],auto[0]>          3752          1          -    Covered
#             bin <auto[0],auto[0],auto[0]>          2115          1          -    Covered
#         Illegal and Ignore Bins:
#             illegal_bin overflow_illegal              0                     -    ZERO
```

```
#     Cross full_cr                                100.00%        100          -    Covered
#         covered/total bins:                            6          6          -
#         missing/total bins:                            0          6          -
#         % Hit:                                   100.00%        100          -
#         Auto, Default and User Defined Bins:
#             bin <auto[1],auto[1],auto[0]>           2111          1          -    Covered
#             bin <auto[0],auto[1],auto[0]>            936          1          -    Covered
#             bin <auto[1],auto[0],auto[1]>           1905          1          -    Covered
#             bin <auto[1],auto[0],auto[0]>           2934          1          -    Covered
#             bin <auto[0],auto[0],auto[1]>            520          1          -    Covered
#             bin <auto[0],auto[0],auto[0]>           1595          1          -    Covered
#         Illegal and Ignore Bins:
#             illegal_bin full_illegal                   0                     -    ZERO
#     Cross empty_cr                               100.00%        100          -    Covered
#         covered/total bins:                            8          8          -
#         missing/total bins:                            0          8          -
#         % Hit:                                   100.00%        100          -
#         Auto, Default and User Defined Bins:
#             bin <auto[1],auto[1],auto[1]>           105          1          -    Covered
#             bin <auto[0],auto[1],auto[1]>           196          1          -    Covered
#             bin <auto[1],auto[0],auto[1]>           238          1          -    Covered
#             bin <auto[0],auto[0],auto[1]>           256          1          -    Covered
#             bin <auto[1],auto[1],auto[0]>          2006          1          -    Covered
#             bin <auto[0],auto[1],auto[0]>           740          1          -    Covered
#             bin <auto[1],auto[0],auto[0]>          4601          1          -    Covered
#             bin <auto[0],auto[0],auto[0]>          1859          1          -    Covered
#     Cross almostfull_cr                          100.00%        100          -    Covered
#         covered/total bins:                            8          8          -
#         missing/total bins:                            0          8          -
#         % Hit:                                   100.00%        100          -
#         Auto, Default and User Defined Bins:
#             bin <auto[1],auto[1],auto[1]>           802          1          -    Covered
#             bin <auto[0],auto[1],auto[1]>           225          1          -    Covered
#             bin <auto[1],auto[0],auto[1]>           377          1          -    Covered
#             bin <auto[0],auto[0],auto[1]>           333          1          -    Covered
#             bin <auto[1],auto[1],auto[0]>          1309          1          -    Covered
#             bin <auto[0],auto[1],auto[0]>           711          1          -    Covered
#             bin <auto[1],auto[0],auto[0]>          4462          1          -    Covered
#             bin <auto[0],auto[0],auto[0]>          1782          1          -    Covered
```

```
#     Cross almostempty_cr                         100.00%        100          -    Covered
#         covered/total bins:                            8          8          -
#         missing/total bins:                            0          8          -
#         % Hit:                                   100.00%        100          -
#         Auto, Default and User Defined Bins:
#             bin <auto[1],auto[1],auto[1]>           380          1          -    Covered
#             bin <auto[0],auto[1],auto[1]>            69          1          -    Covered
#             bin <auto[1],auto[0],auto[1]>           368          1          -    Covered
#             bin <auto[0],auto[0],auto[1]>           183          1          -    Covered
#             bin <auto[1],auto[1],auto[0]>          1731          1          -    Covered
#             bin <auto[0],auto[1],auto[0]>           867          1          -    Covered
#             bin <auto[1],auto[0],auto[0]>          4471          1          -    Covered
#             bin <auto[0],auto[0],auto[0]>          1932          1          -    Covered
#     Cross underflow_cr                           100.00%        100          -    Covered
#         covered/total bins:                            6          6          -
#         missing/total bins:                            0          6          -
#         % Hit:                                   100.00%        100          -
#         Auto, Default and User Defined Bins:
#             bin <auto[1],auto[1],auto[1]>           154          1          -    Covered
#             bin <auto[1],auto[1],auto[0]>          1957          1          -    Covered
#             bin <auto[0],auto[1],auto[1]>            68          1          -    Covered
#             bin <auto[0],auto[1],auto[0]>           868          1          -    Covered
#             bin <auto[1],auto[0],auto[0]>          4839          1          -    Covered
#             bin <auto[0],auto[0],auto[0]>          2115          1          -    Covered
#         Illegal and Ignore Bins:
#             illegal bin underflow illegal              0                     -    ZERO
```