ECE 560 ASSERTIONS BASED VERIFICATION
HOMEWORK – 3 TEAM 16
SIMPLE 2x2 ROUTER

Name: Poojith Pogarthi PSU ID: 958547659
Name: Sri Chandana Reddy Vanukuri PSU ID: 989166262

After carefully going through the specification document, we encountered that there is an ambiguity in the specification document itself.

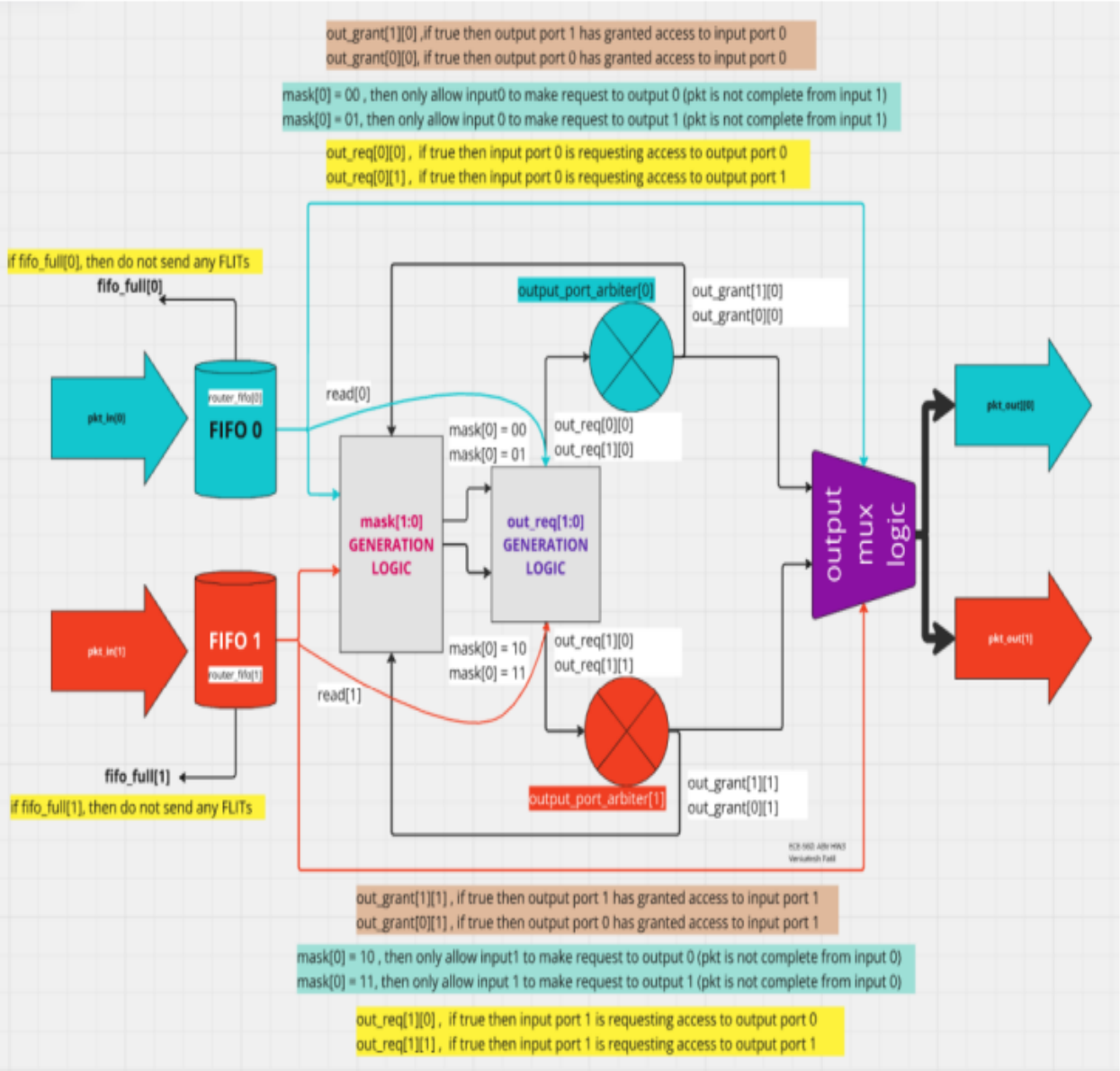
We are considering the specification 1 rules and starting the verification process for the simple 2x2 router design.

Specification 1:

Mask Generation Logic (prevent a complete packet has is observed at output after the grant is give, block new request from grant)

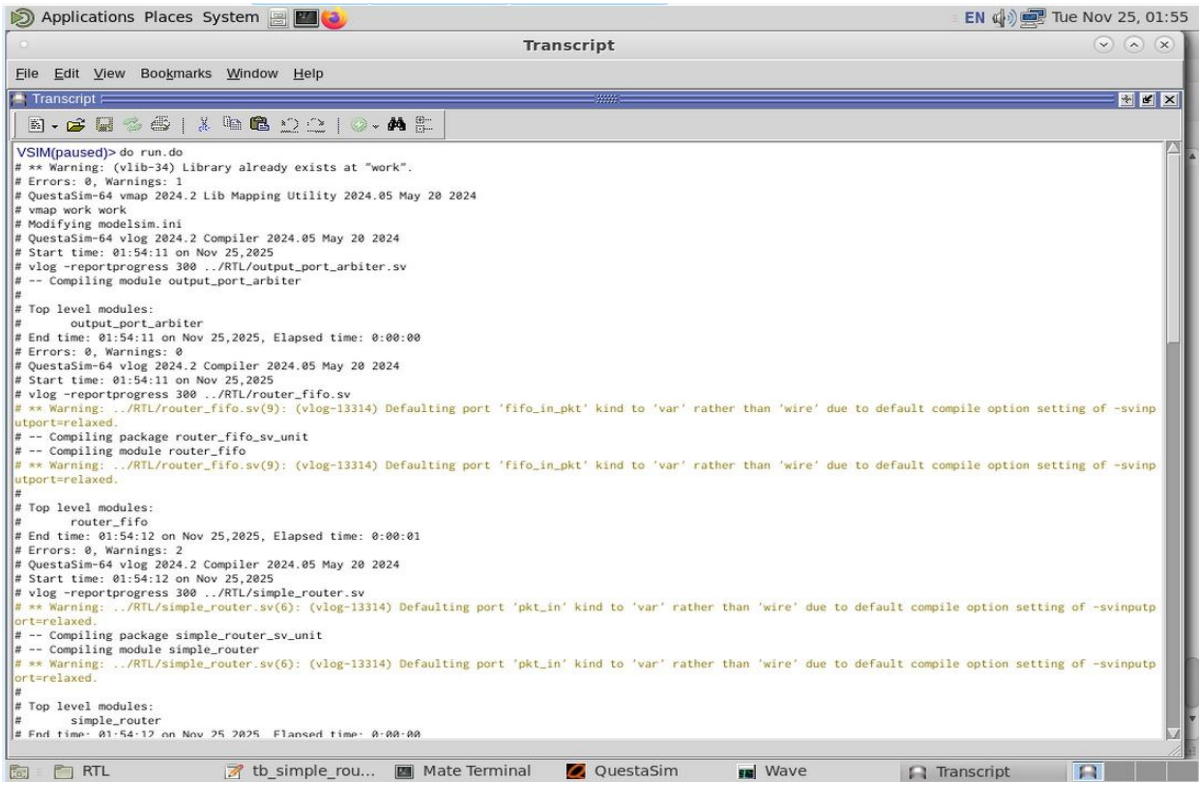
- if an output port arbiter gives grant to an input, then until we have sent out the entire packet (head flit, middle flit and tail flit) out from the same input port, we should not give grant to another input port
- Example: if out_grant[0][1] is 1 , that means output port 0 gave grant to input port 1, then the out_grant[0][1] should again be 1 until we have sent pkt_out[0].tail == 1
- This is implemented in RTL, using two bit mask signals mask[0] and mask[1]
    - if mask[0] = 2'b11, that means both the inputs are allowed to make a request to output port 0 (out_req[0][0] and out_req[0][1] may both be high
    - if mask[1] = 2'b11, that means both the inputs are allowed to make request to output port 1 (out_req[1][0] and out_req[1][1] may both be high
- if we give grant to an input from a output port, and the tail bit of the data coming from input(fifo_out_pkt[0].tail) is 0, then next cycle, we will block another input to send out_req to output port
- Example: if out_grant[0][1] == 1 (output 0 gave grant to input 1) and if fifo_out_pkt[1].tail == 0, then in next cycle, mask[0] = 2'b10 (we will only allow input 1 to make request to output 0 (out_req[0][1] may be high, but out_req[1][1] must be 0

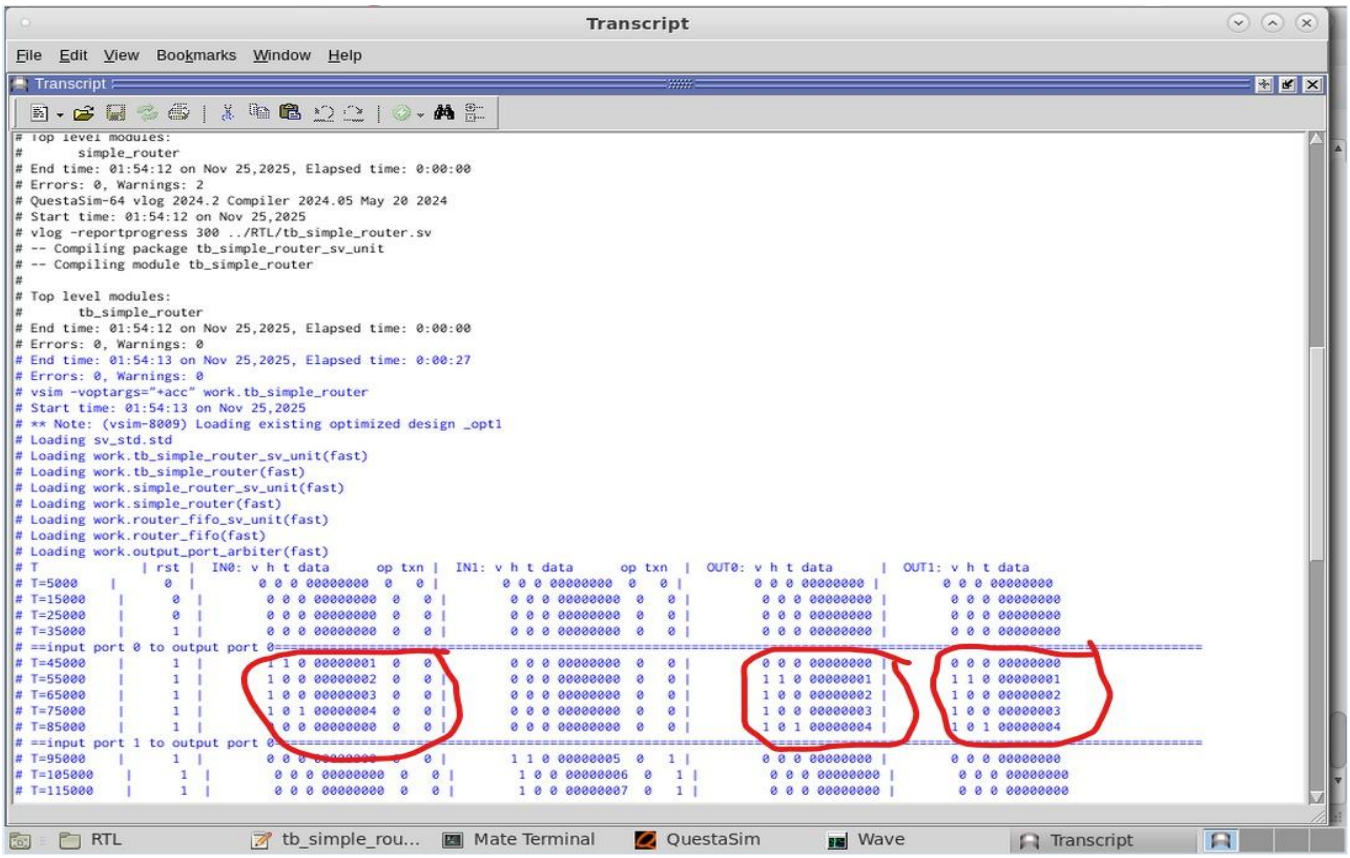
Specification 2:



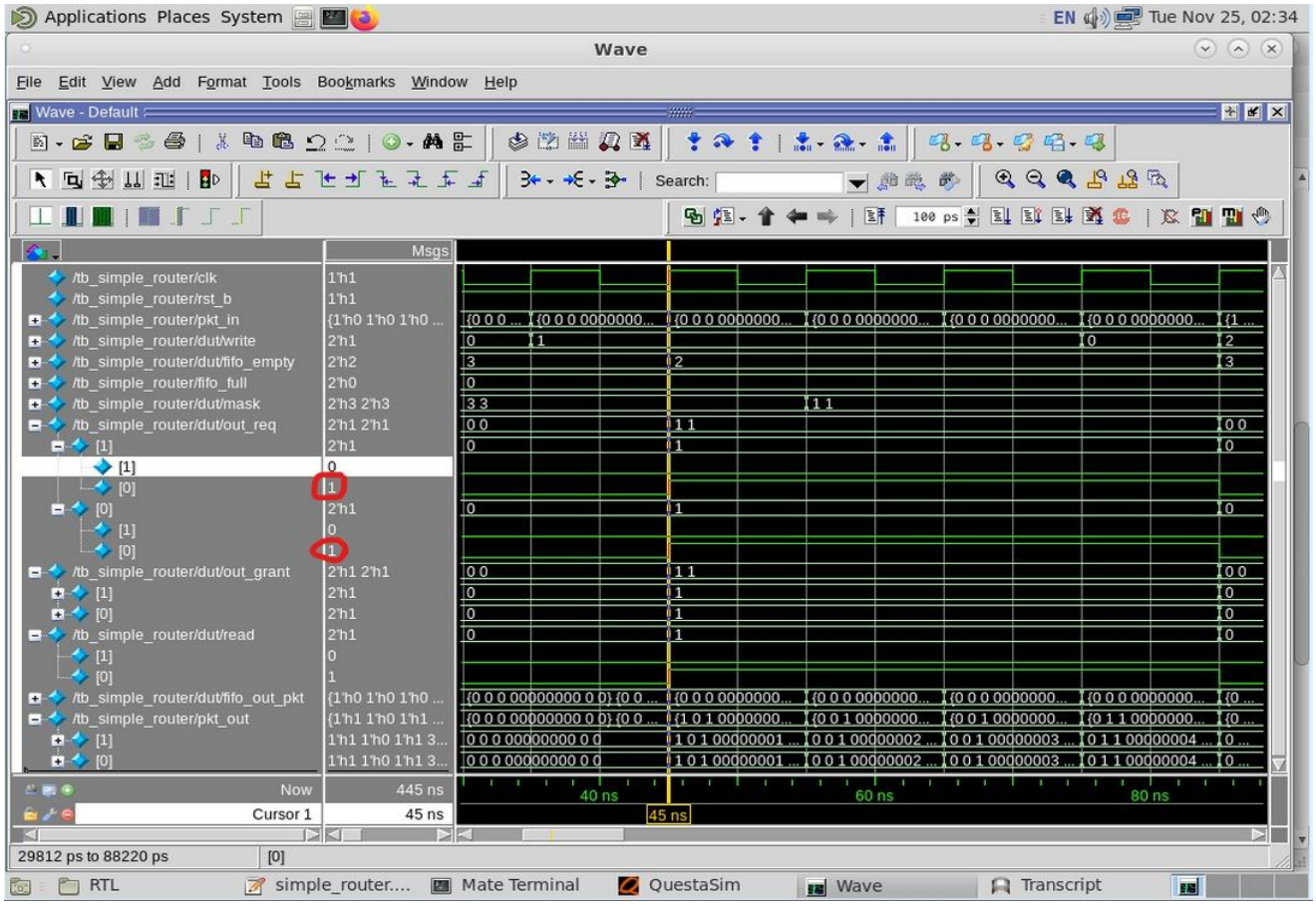
Part -1 Dynamic Simulation in QuestaSim

1. After reviewing the given RTL files in the simulation folder as mentioned in the README file, we included the design_includes.h file, which contains the struct packet, in all the required design files.
2. After successfully compiling the design with a simple testbench, we observed 0 errors and 5 warnings. These warnings indicate:
   - The work library already exists because the simulation was run more than once. This warning will not appear during the first run.
   - The remaining four warnings are related to signal type clarification. Since the ports were not explicitly declared as wire, the tool treated them as var by default. This happened because they are used in procedural blocks and the compile option -svinputport=relaxed is enabled.




Transcript
File Edit View Bookmarks Window Help
Transcript
Top level modules:
simple_router
End time: 01:54:12 on Nov 25,2025, Elapsed time: 0:00:00
Errors: 0, Warnings: 2
QuestaSim-64 vlog 2024.2 Compiler 2024.05 May 20 2024
Start time: 01:54:12 on Nov 25,2025
vlog -reportprogress 300 ../RTL/tb_simple_router.sv
-- Compiling package tb_simple_router_sv_unit
-- Compiling module tb_simple_router
#
Top level modules:
tb_simple_router
End time: 01:54:12 on Nov 25,2025, Elapsed time: 0:00:00
Errors: 0, Warnings: 0
End time: 01:54:13 on Nov 25,2025, Elapsed time: 0:00:27
Errors: 0, Warnings: 0
vsim -voptargs="+acc" work.tb_simple_router
Start time: 01:54:13 on Nov 25,2025
** Note: (vsim-8009) Loading existing optimized design _opt1
Loading sv_std.std
Loading work.tb_simple_router_sv_unit(fast)
Loading work.tb_simple_router(fast)
Loading work.simple_router_sv_unit(fast)
Loading work.simple_router(fast)
Loading work.router_fifo_sv_unit(fast)
Loading work.router_fifo(fast)
Loading work.output_port_arbiter(fast)
T | rst | IN0: v h t data op txn | IN1: v h t data op txn | OUT0: v h t data | OUT1: v h t data
T=5000 | 0 | 0 0 0 00000000 0 0 | 0 0 0 00000000 0 0 | 0 0 0 00000000 | 0 0 0 00000000
T=15000 | 0 | 0 0 0 00000000 0 0 | 0 0 0 00000000 0 0 | 0 0 0 00000000 | 0 0 0 00000000
T=25000 | 0 | 0 0 0 00000000 0 0 | 0 0 0 00000000 0 0 | 0 0 0 00000000 | 0 0 0 00000000
T=35000 | 1 | 0 0 0 00000000 0 0 | 0 0 0 00000000 0 0 | 0 0 0 00000000 | 0 0 0 00000000
==input port 0 to output port 0==========
T=45000 | 1 | 1 1 0 00000001 0 0 | 0 0 0 00000000 0 0 | 0 0 0 00000000 | 0 0 0 00000000
T=55000 | 1 | 1 0 0 00000002 0 0 | 0 0 0 00000000 0 0 | 1 1 0 00000001 | 1 1 0 00000001
T=65000 | 1 | 1 0 0 00000003 0 0 | 0 0 0 00000000 0 0 | 1 0 0 00000002 | 1 0 0 00000002
T=75000 | 1 | 1 0 1 00000004 0 0 | 0 0 0 00000000 0 0 | 1 0 0 00000003 | 1 0 0 00000003
T=85000 | 1 | 0 0 0 00000000 0 0 | 0 0 0 00000000 0 0 | 1 0 1 00000004 | 1 0 1 00000004
==input port 1 to output port 0==========
T=95000 | 1 | 0 0 0 00000000 0 0 | 1 1 0 00000005 0 1 | 0 0 0 00000000 | 0 0 0 00000000
T=105000 | 1 | 0 0 0 00000000 0 0 | 1 0 0 00000006 0 1 | 0 0 0 00000000 | 0 0 0 00000000
T=115000 | 1 | 0 0 0 00000000 0 0 | 1 0 0 00000007 0 1 | 0 0 0 00000000 | 0 0 0 00000000
Applications Places System
EN Tue Nov 25, 01:57
Transcript
File Edit View Bookmarks Window Help
Transcript
T=65000 | 1 | 1 0 0 00000003 0 0 | 0 0 0 00000000 0 0 | 1 0 0 00000002 | 1 0 0 00000002
T=75000 | 1 | 1 0 1 00000004 0 0 | 0 0 0 00000000 0 0 | 1 0 0 00000003 | 1 0 0 00000003
T=85000 | 1 | 0 0 0 00000000 0 0 | 0 0 0 00000000 0 0 | 1 0 1 00000004 | 1 0 1 00000004
==input port 1 to output port 0==========
T=95000 | 1 | 0 0 0 00000000 0 0 | 1 1 0 00000005 0 1 | 0 0 0 00000000 | 0 0 0 00000000
T=105000 | 1 | 0 0 0 00000000 0 0 | 1 0 0 00000006 0 1 | 0 0 0 00000000 | 0 0 0 00000000
T=115000 | 1 | 0 0 0 00000000 0 0 | 1 0 0 00000007 0 1 | 0 0 0 00000000 | 0 0 0 00000000
T=125000 | 1 | 0 0 0 00000000 0 0 | 1 0 1 00000008 0 1 | 0 0 0 00000000 | 0 0 0 00000000
T=135000 | 1 | 0 0 0 00000000 0 0 | 0 0 0 00000000 0 1 | 0 0 0 00000000 | 0 0 0 00000000
==input port 0 to output port 1==========
T=145000 | 1 | 1 1 0 00000001 1 0 | 0 0 0 00000000 0 1 | 0 0 0 00000000 | 0 0 0 00000000
T=155000 | 1 | 1 0 0 00000002 1 0 | 0 0 0 00000000 0 1 | 0 0 0 00000000 | 0 0 0 00000000
T=165000 | 1 | 1 0 1 00000004 1 0 | 0 0 0 00000000 0 1 | 0 0 0 00000000 | 0 0 0 00000000
T=175000 | 1 | 0 0 0 00000000 1 0 | 0 0 0 00000000 0 1 | 0 0 0 00000000 | 0 0 0 00000000
==input port 1 to output port 1==========
T=185000 | 1 | 0 0 0 00000000 1 0 | 1 1 0 00000001 1 1 | 0 0 0 00000000 | 0 0 0 00000000
T=195000 | 1 | 0 0 0 00000000 1 0 | 1 0 0 00000002 1 1 | 0 0 0 00000000 | 0 0 0 00000000
T=205000 | 1 | 0 0 0 00000000 1 0 | 1 0 1 00000004 1 1 | 0 0 0 00000000 | 0 0 0 00000000
T=215000 | 1 | 0 0 0 00000000 1 0 | 0 0 0 00000000 1 1 | 0 0 0 00000000 | 0 0 0 00000000
==input port 1 to output port 1,input port 0 to output port 0==========
T=225000 | 1 | 1 1 0 00000001 0 0 | 1 1 0 00000001 1 1 | 0 0 0 00000000 | 0 0 0 00000000
T=235000 | 1 | 1 0 0 00000001 0 0 | 1 1 0 00000002 1 1 | 0 0 0 00000000 | 0 0 0 00000000
T=245000 | 1 | 1 0 0 00000001 0 0 | 1 0 0 00000003 1 1 | 0 0 0 00000000 | 0 0 0 00000000
T=255000 | 1 | 1 0 1 00000001 0 0 | 1 0 1 00000004 1 1 | 0 0 0 00000000 | 0 0 0 00000000
T=265000 | 1 | 0 0 0 00000000 0 0 | 0 0 0 00000000 1 1 | 0 0 0 00000000 | 0 0 0 00000000
==input port 1 to output port 0,input port 0 to output port 0==========
T=275000 | 1 | 1 1 0 00000001 0 0 | 1 1 0 00000001 0 1 | 0 0 0 00000000 | 0 0 0 00000000
T=285000 | 1 | 1 0 0 00000001 0 0 | 1 1 0 00000002 0 1 | 0 0 0 00000000 | 0 0 0 00000000
T=295000 | 1 | 1 0 0 00000001 0 0 | 1 0 0 00000003 0 1 | 0 0 0 00000000 | 0 0 0 00000000
T=305000 | 1 | 1 0 1 00000001 0 0 | 1 0 1 00000004 0 1 | 0 0 0 00000000 | 0 0 0 00000000
T=315000 | 1 | 0 0 0 00000000 0 0 | 0 0 0 00000000 0 1 | 0 0 0 00000000 | 0 0 0 00000000
==input port 1 to output port 1,input port 0 to output port 1==========
T=325000 | 1 | 1 1 0 00000001 1 0 | 1 1 0 00000001 1 1 | 0 0 0 00000000 | 0 0 0 00000000
T=335000 | 1 | 1 0 0 00000001 1 0 | 1 1 0 00000002 1 1 | 0 0 0 00000000 | 0 0 0 00000000
T=345000 | 1 | 1 0 0 00000001 1 0 | 1 0 0 00000003 1 1 | 0 0 0 00000000 | 0 0 0 00000000
T=355000 | 1 | 1 0 1 00000001 1 0 | 1 0 1 00000004 1 1 | 0 0 0 00000000 | 0 0 0 00000000
T=365000 | 1 | 0 0 0 00000000 1 0 | 1 1 0 00000001 1 1 | 0 0 0 00000000 | 0 0 0 00000000
T=375000 | 1 | 0 0 0 00000000 1 0 | 1 0 1 00000004 1 1 | 0 0 0 00000000 | 0 0 0 00000000
T=385000 | 1 | 0 0 0 00000000 1 0 | 1 0 0 00000004 1 1 | 0 0 0 00000000 | 0 0 0 00000000
T=395000 | 1 | 0 0 0 00000000 1 0 | 1 0 1 00000004 1 1 | 0 0 0 00000000 | 0 0 0 00000000
RTL
tb_simple_rou...
Mate Terminal
QuestaSim
Wave
Transcript

From the compilation results, we observed that transactions were not flowing correctly from the input ports to the output ports as per the required specifications. This indicated the presence of a bug in the design.



The issue was that both output arbiters were issuing grants to input 0, since input 0 had the highest priority for both outputs. This behavior was incorrect and pointed to a problem in the out_req logic at lines 42 and 43.

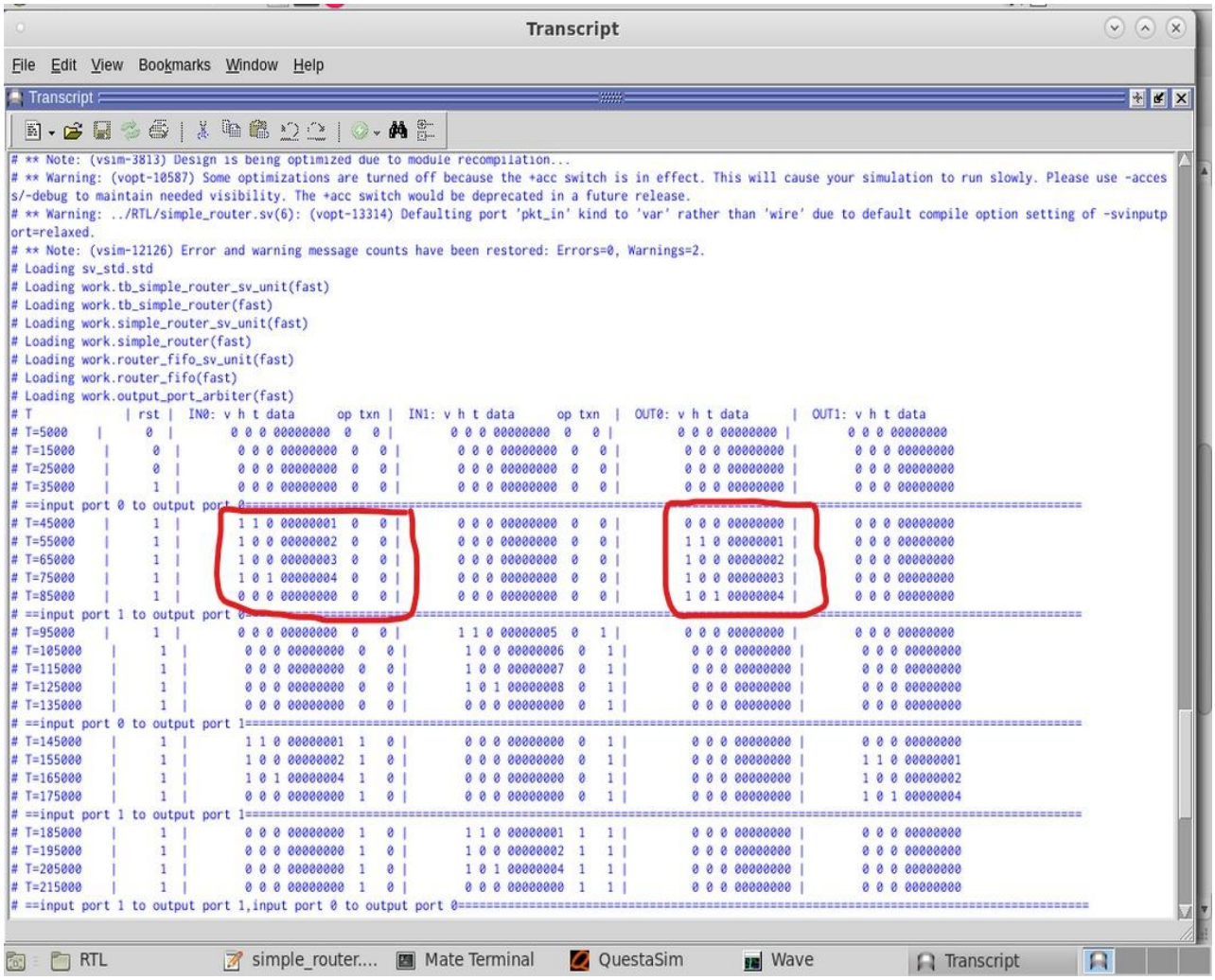
Before Bug Fix:

```
40
41 assign out_req[0][0] = !fifo_empty[0] ? (fifo_out_pkt[0].output_port_num == 0) && mask[0][0] : 1'b0;
42 assign out_req[0][1] = !fifo_empty[1] ? (fifo_out_pkt[1].output_port_num == 1) && mask[0][1] : 1'b0;
43 assign out_req[1][0] = !fifo_empty[0] ? (fifo_out_pkt[0].output_port_num == 0) && mask[1][0] : 1'b0;
44 assign out_req[1][1] = !fifo_empty[1] ? (fifo_out_pkt[1].output_port_num == 1) && mask[1][1] : 1'b0;
45
```

After Bug Fix:

```
40
41 assign out_req[0][0] = !fifo_empty[0] ? (fifo_out_pkt[0].output_port_num == 0) && mask[0][0] : 1'b0;
42 assign out_req[0][1] = !fifo_empty[1] ? (fifo_out_pkt[1].output_port_num == 0) && mask[0][1] : 1'b0;
43 assign out_req[1][0] = !fifo_empty[0] ? (fifo_out_pkt[0].output_port_num == 1) && mask[1][0] : 1'b0;
44 assign out_req[1][1] = !fifo_empty[1] ? (fifo_out_pkt[1].output_port_num == 1) && mask[1][1] : 1'b0;
45
```

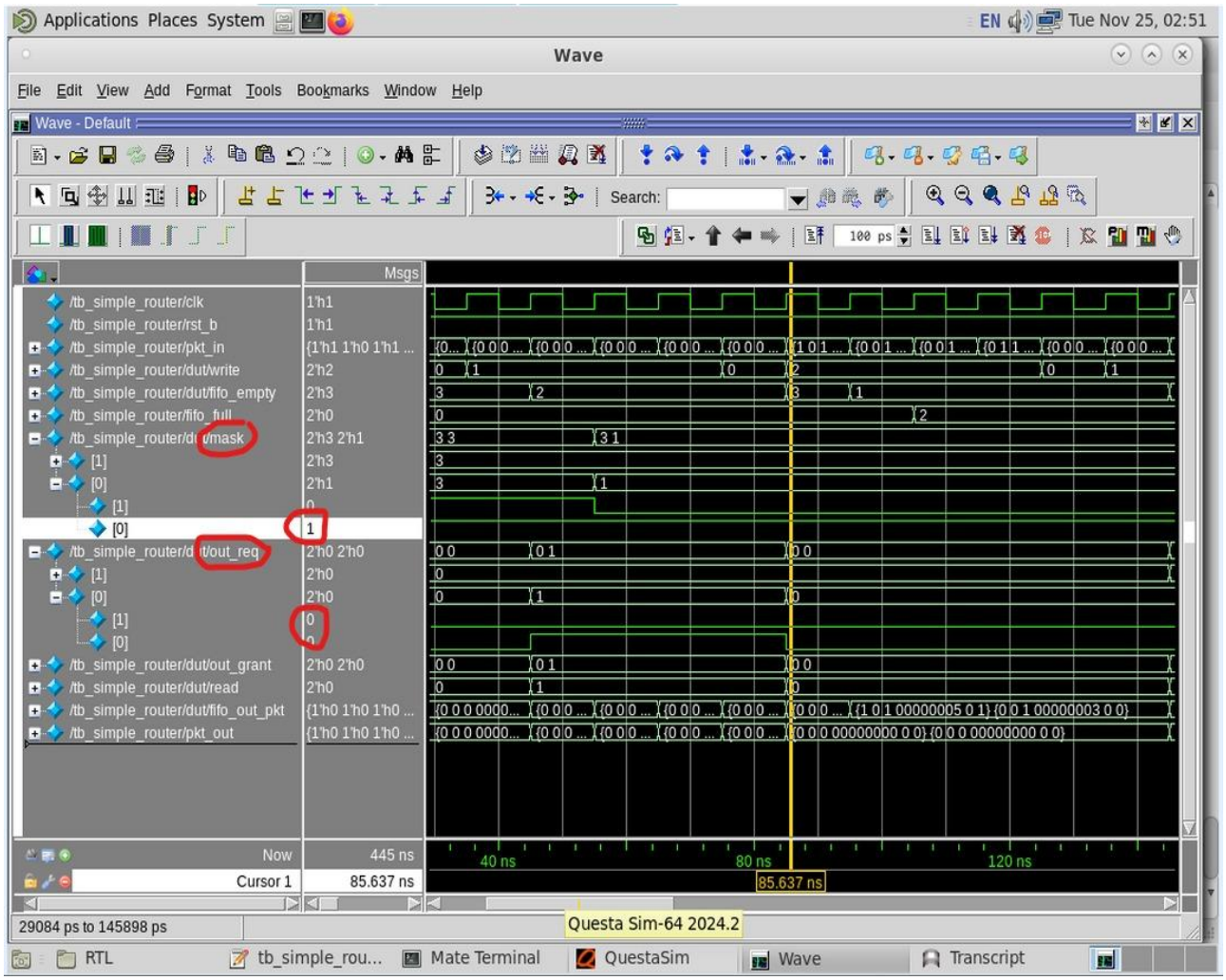
After correcting this logic in the simple_router.sv design file, the routing behavior worked as expected, and input 0 was correctly connected to output 0.



We still observed an issue with the transaction flowing from input 1 to output 0.

After all FLITS of the first transaction from input 0 to output 0 were processed, the mask value became mask[0] = 01. This meant that only input 0 was allowed to request access to output 0, while input 1 was blocked.

However, according to the expected behavior, both inputs should be able to request access to output 0. To fix this issue, we modified the RTL design by changing the mask value from mask[0] = 01 to mask[0] = 11, allowing both input 0 and input 1 to request access to output 0 as intended.



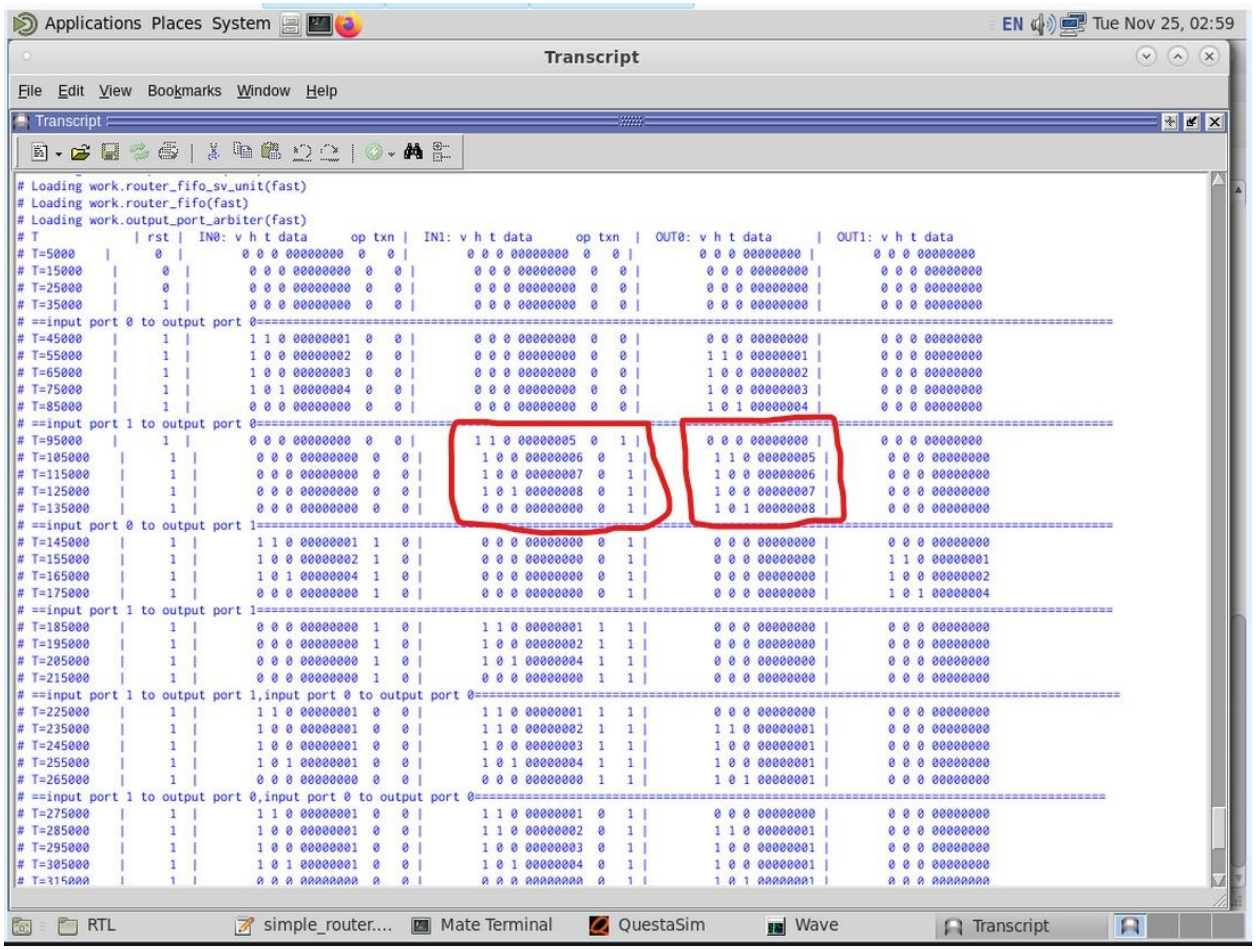
Before Bug Fix:

```
61    else begin
62    if((out_grant[0][0] && fifo_out_pkt[0].tail) || (out_grant[0][1] && fifo_out_pkt[1].tail)) begin
63        mask[0] <= 2'h1;
64    end
```

After Bug Fix:

```
61    else begin
62    if((out_grant[0][0] && fifo_out_pkt[0].tail) || (out_grant[0][1] && fifo_out_pkt[1].tail)) begin
63        mask[0] <= 2'h3;
```

After fixing this bug, packet successfully transacts from input 1 to output 0.



We now see a similar issue with the input 1 masking logic. After a transaction is received, the mask value for output 1 always gives priority to input 0. Because of this, the transaction from input 1 cannot reach output 1 as expected.

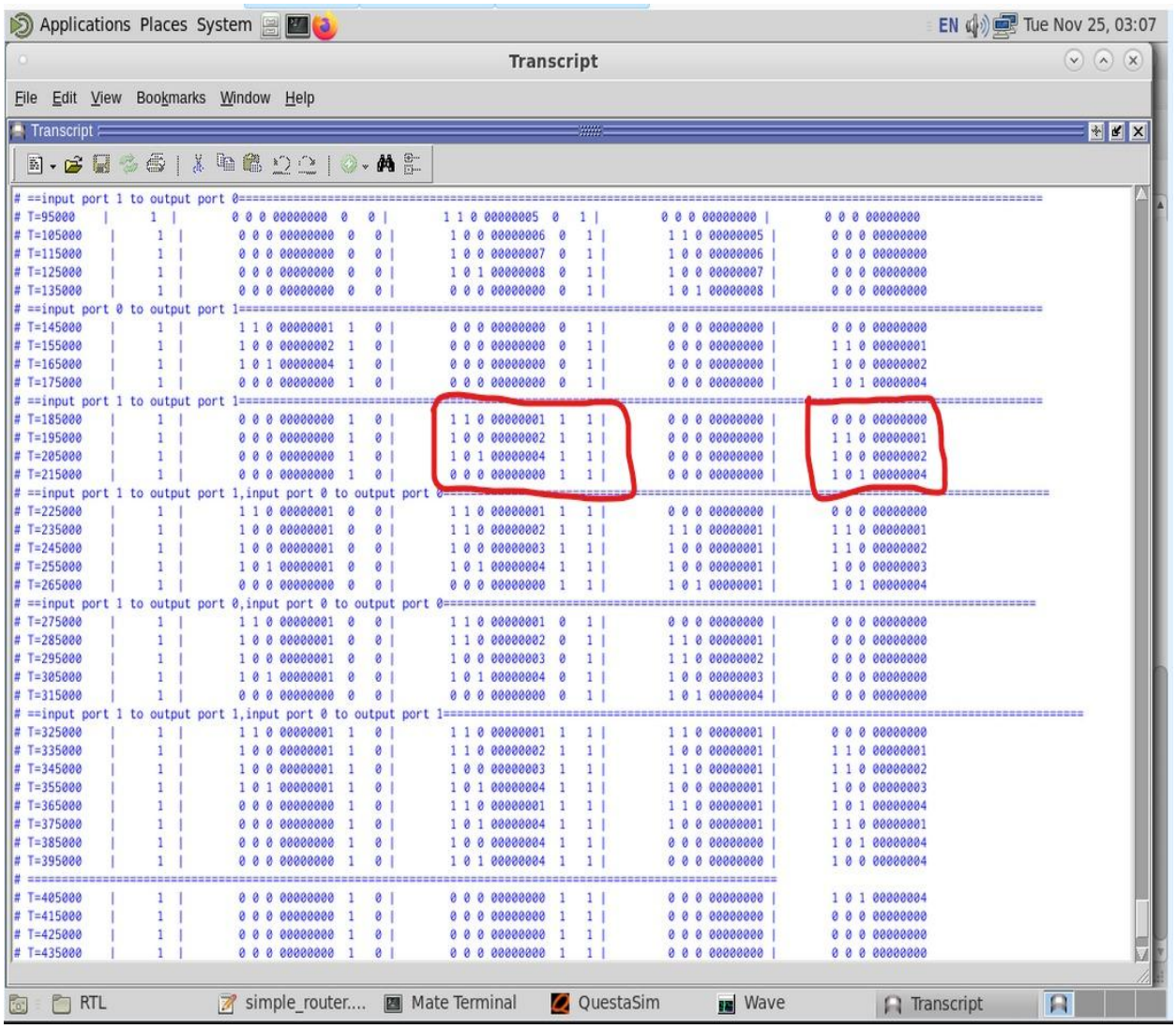
This indicates that the masking logic for output 1 is not being updated correctly, and input 1 is being unfairly blocked while input 0 continues to receive priority. This needs to be corrected so that both inputs are given proper and fair access to output 1 based on the specification rules.

Before Bug Fix:

```
71
72      if((out_grant[1][0] && fifo_out_pkt[0].tail) || (out_grant[1][1] && fifo_out_pkt[1].tail)) begin
73          mask[1] <= 2'h1;
74      end
```

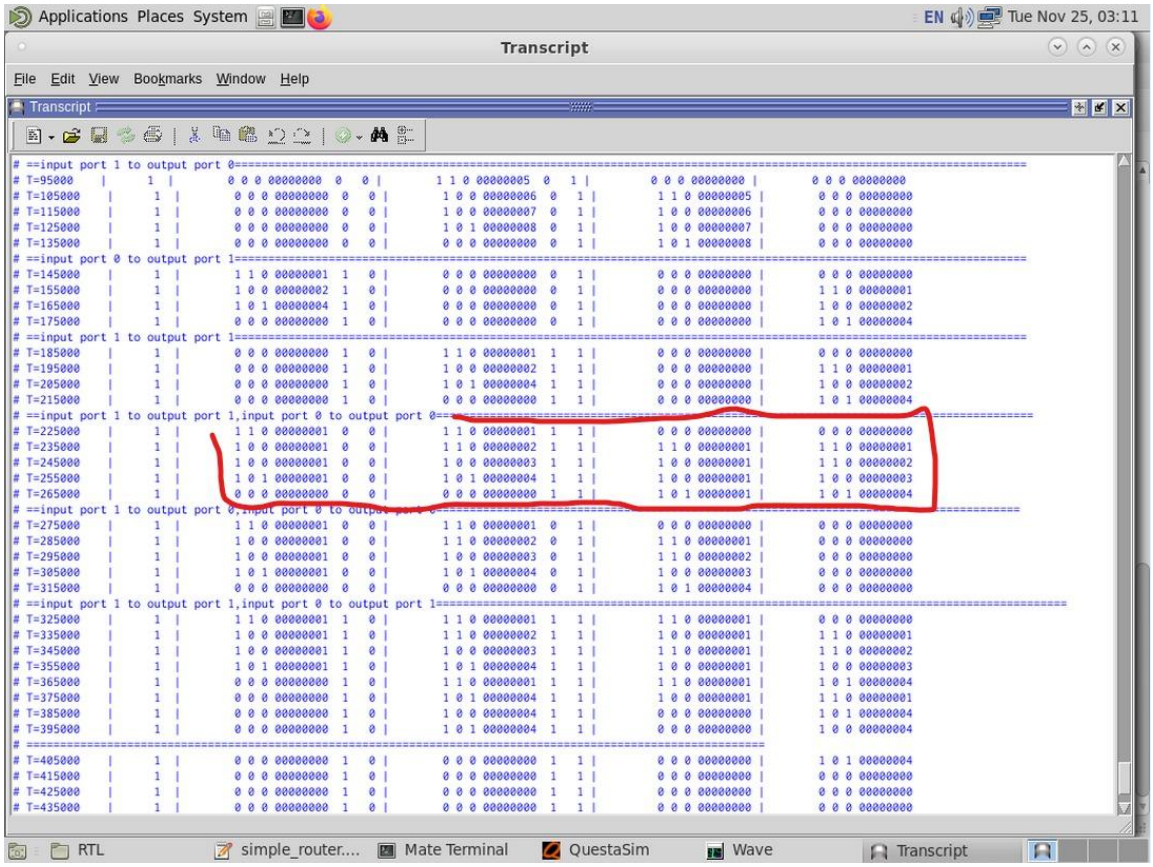
After Bug Fix:

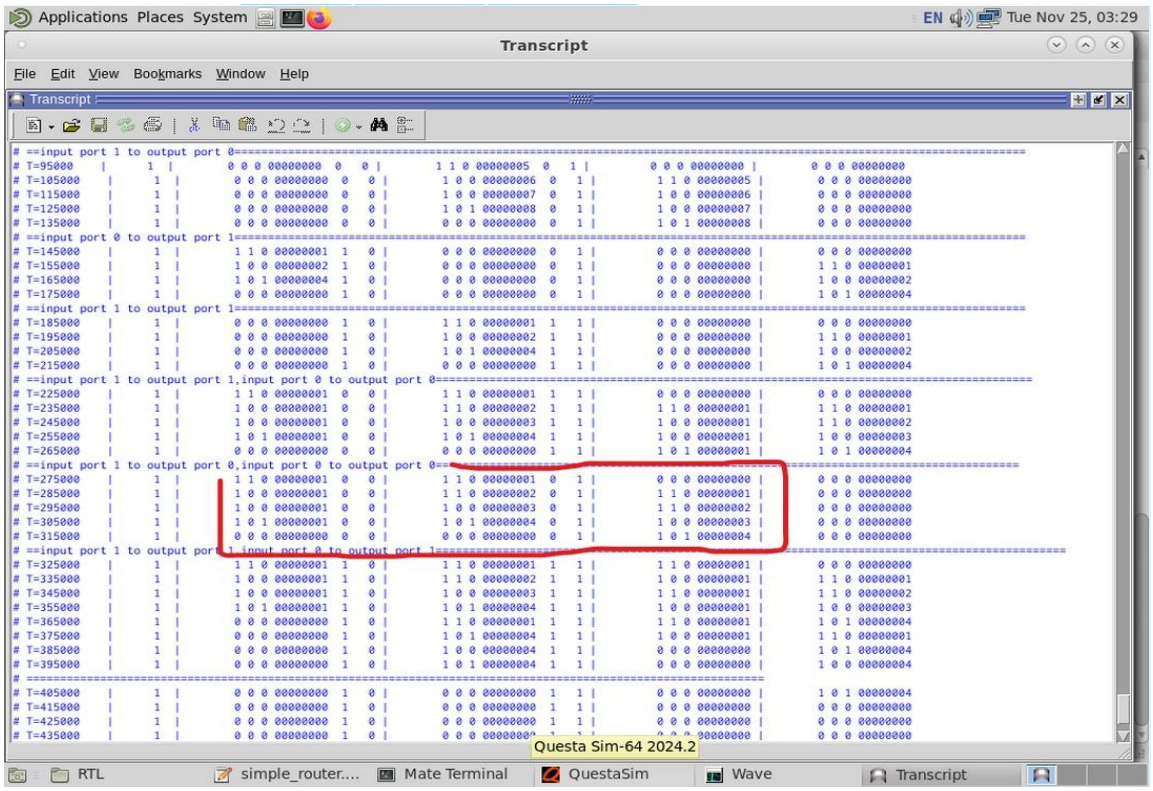
```
71
72      if((out_grant[1][0] && fifo_out_pkt[0].tail) || (out_grant[1][1] && fifo_out_pkt[1].tail)) begin
73          mask[1] <= 2'h3;
74      end
```



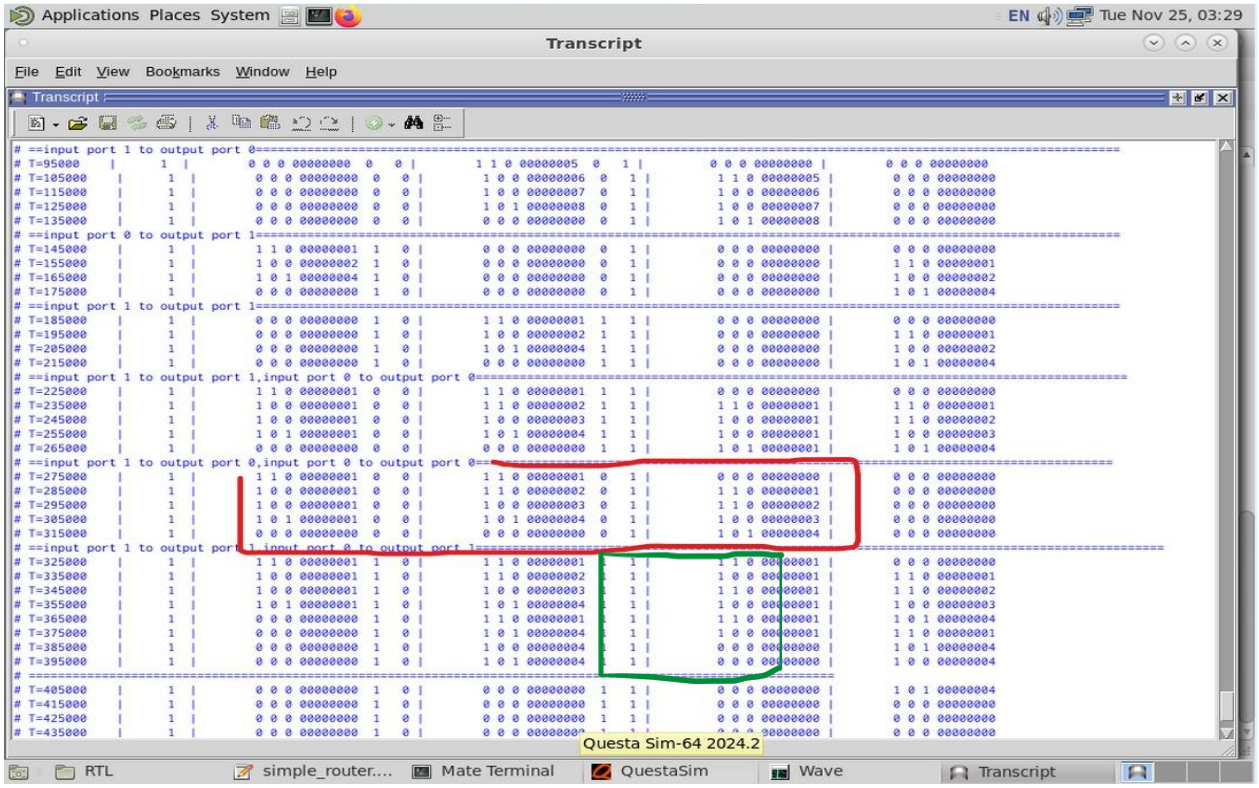
We have tested few more features, including:

- We are sending transactions from both inputs to corresponding outputs,
  Input[0] => Output[0]
  Input[1] => Output[1]
- We are sending transactions from both inputs to a single output,
  Input[1] and Input[0] => Output[0]
  The input port given priority will go to the output.
- FIFO Issue


Applications Places System
EN Tue Nov 25, 03:11
Transcript
File Edit View Bookmarks Window Help
Transcript
==input port 1 to output port 0==========
T=95000 | 1 | 0 0 0 00000000 0 0 | 1 1 0 00000005 0 1 | 0 0 0 00000000 | 0 0 0 00000000
T=105000 | 1 | 0 0 0 00000000 0 0 | 1 0 0 00000006 0 1 | 1 1 0 00000005 | 0 0 0 00000000
T=115000 | 1 | 0 0 0 00000000 0 0 | 1 0 0 00000007 0 1 | 1 0 0 00000006 | 0 0 0 00000000
T=125000 | 1 | 0 0 0 00000000 0 0 | 1 0 1 00000008 0 1 | 1 0 0 00000007 | 0 0 0 00000000
T=135000 | 1 | 0 0 0 00000000 0 0 | 0 0 0 00000000 0 1 | 1 0 1 00000008 | 0 0 0 00000000
==input port 0 to output port 1==========
T=145000 | 1 | 1 1 0 00000001 1 0 | 0 0 0 00000000 0 1 | 0 0 0 00000000 | 0 0 0 00000000
T=155000 | 1 | 1 0 0 00000002 1 0 | 0 0 0 00000000 0 1 | 0 0 0 00000000 | 1 1 0 00000001
T=165000 | 1 | 1 0 1 00000004 1 0 | 0 0 0 00000000 0 1 | 0 0 0 00000000 | 1 0 0 00000002
T=175000 | 1 | 0 0 0 00000000 1 0 | 0 0 0 00000000 0 1 | 0 0 0 00000000 | 1 0 1 00000004
==input port 1 to output port 1==========
T=185000 | 1 | 0 0 0 00000000 1 0 | 1 1 0 00000001 1 1 | 0 0 0 00000000 | 0 0 0 00000000
T=195000 | 1 | 0 0 0 00000000 1 0 | 1 0 0 00000002 1 1 | 0 0 0 00000000 | 1 1 0 00000001
T=205000 | 1 | 0 0 0 00000000 1 0 | 1 0 1 00000004 1 1 | 0 0 0 00000000 | 1 0 0 00000002
T=215000 | 1 | 0 0 0 00000000 1 0 | 0 0 0 00000000 1 1 | 0 0 0 00000000 | 1 0 1 00000004
==input port 1 to output port 1,input port 0 to output port 0==========
T=225000 | 1 | 1 1 0 00000001 0 0 | 1 1 0 00000001 1 1 | 0 0 0 00000000 | 0 0 0 00000000
T=235000 | 1 | 1 0 0 00000001 0 0 | 1 1 0 00000002 1 1 | 1 1 0 00000001 | 1 1 0 00000001
T=245000 | 1 | 1 0 0 00000001 0 0 | 1 0 0 00000003 1 1 | 1 0 0 00000001 | 1 1 0 00000002
T=255000 | 1 | 1 0 1 00000001 0 0 | 1 0 1 00000004 1 1 | 1 0 0 00000001 | 1 0 0 00000003
T=265000 | 1 | 0 0 0 00000000 0 0 | 0 0 0 00000000 1 1 | 1 0 1 00000001 | 1 0 1 00000004
==input port 1 to output port 0,input port 0 to output port 0==========
T=275000 | 1 | 1 1 0 00000001 0 0 | 1 1 0 00000001 0 1 | 0 0 0 00000000 | 0 0 0 00000000
T=285000 | 1 | 1 0 0 00000001 0 0 | 1 1 0 00000002 0 1 | 1 1 0 00000001 | 0 0 0 00000000
T=295000 | 1 | 1 0 0 00000001 0 0 | 1 0 0 00000003 0 1 | 1 0 0 00000001 | 0 0 0 00000000
T=305000 | 1 | 1 0 1 00000001 0 0 | 1 0 1 00000004 0 1 | 1 0 0 00000001 | 0 0 0 00000000
T=315000 | 1 | 0 0 0 00000000 0 0 | 0 0 0 00000000 0 1 | 1 0 1 00000004 | 0 0 0 00000000
==input port 1 to output port 1,input port 0 to output port 1==========
T=325000 | 1 | 1 1 0 00000001 1 0 | 1 1 0 00000001 1 1 | 1 1 0 00000001 | 0 0 0 00000000
T=335000 | 1 | 1 0 0 00000001 1 0 | 1 1 0 00000002 1 1 | 1 0 0 00000001 | 1 1 0 00000001
T=345000 | 1 | 1 0 0 00000001 1 0 | 1 0 0 00000003 1 1 | 1 1 0 00000001 | 1 1 0 00000002
T=355000 | 1 | 1 0 1 00000001 1 0 | 1 0 1 00000004 1 1 | 1 0 0 00000001 | 1 0 0 00000003
T=365000 | 1 | 0 0 0 00000000 1 0 | 1 1 0 00000001 1 1 | 1 1 0 00000001 | 1 0 1 00000004
T=375000 | 1 | 0 0 0 00000000 1 0 | 1 0 1 00000004 1 1 | 1 0 0 00000001 | 1 1 0 00000001
T=385000 | 1 | 0 0 0 00000000 1 0 | 1 0 0 00000004 1 1 | 0 0 0 00000000 | 1 0 1 00000004
T=395000 | 1 | 0 0 0 00000000 1 0 | 1 0 1 00000004 1 1 | 0 0 0 00000000 | 1 0 0 00000004
==========
T=405000 | 1 | 0 0 0 00000000 1 0 | 0 0 0 00000000 1 1 | 0 0 0 00000000 | 1 0 1 00000004
T=415000 | 1 | 0 0 0 00000000 1 0 | 0 0 0 00000000 1 1 | 0 0 0 00000000 | 0 0 0 00000000
T=425000 | 1 | 0 0 0 00000000 1 0 | 0 0 0 00000000 1 1 | 0 0 0 00000000 | 0 0 0 00000000
T=435000 | 1 | 0 0 0 00000000 1 0 | 0 0 0 00000000 1 1 | 0 0 0 00000000 | 0 0 0 00000000
RTL
simple_router....
Mate Terminal
QuestaSim
Wave
Transcript


Applications Places System
EN Tue Nov 25, 03:29
Transcript
File Edit View Bookmarks Window Help
Transcript
==input port 1 to output port 0==========
T=95000 | 1 | 0 0 0 00000000 0 0 | 1 1 0 00000005 0 1 | 0 0 0 00000000 | 0 0 0 00000000
T=105000 | 1 | 0 0 0 00000000 0 0 | 1 0 0 00000006 0 1 | 1 1 0 00000005 | 0 0 0 00000000
T=115000 | 1 | 0 0 0 00000000 0 0 | 1 0 0 00000007 0 1 | 1 0 0 00000006 | 0 0 0 00000000
T=125000 | 1 | 0 0 0 00000000 0 0 | 1 0 1 00000008 0 1 | 1 0 0 00000007 | 0 0 0 00000000
T=135000 | 1 | 0 0 0 00000000 0 0 | 0 0 0 00000000 0 1 | 1 0 1 00000008 | 0 0 0 00000000
==input port 0 to output port 1==========
T=145000 | 1 | 1 1 0 00000001 1 0 | 0 0 0 00000000 0 1 | 0 0 0 00000000 | 0 0 0 00000000
T=155000 | 1 | 1 0 0 00000002 1 0 | 0 0 0 00000000 0 1 | 0 0 0 00000000 | 1 1 0 00000001
T=165000 | 1 | 1 0 1 00000004 1 0 | 0 0 0 00000000 0 1 | 0 0 0 00000000 | 1 0 0 00000002
T=175000 | 1 | 0 0 0 00000000 1 0 | 0 0 0 00000000 0 1 | 0 0 0 00000000 | 1 0 1 00000004
==input port 1 to output port 1==========
T=185000 | 1 | 0 0 0 00000000 1 0 | 1 1 0 00000001 1 1 | 0 0 0 00000000 | 0 0 0 00000000
T=195000 | 1 | 0 0 0 00000000 1 0 | 1 0 0 00000002 1 1 | 0 0 0 00000000 | 1 1 0 00000001
T=205000 | 1 | 0 0 0 00000000 1 0 | 1 0 1 00000004 1 1 | 0 0 0 00000000 | 1 0 0 00000002
T=215000 | 1 | 0 0 0 00000000 1 0 | 0 0 0 00000000 1 1 | 0 0 0 00000000 | 1 0 1 00000004
==input port 1 to output port 1,input port 0 to output port 0==========
T=225000 | 1 | 1 1 0 00000001 0 0 | 1 1 0 00000001 1 1 | 0 0 0 00000000 | 0 0 0 00000000
T=235000 | 1 | 1 0 0 00000001 0 0 | 1 1 0 00000002 1 1 | 1 1 0 00000001 | 1 1 0 00000001
T=245000 | 1 | 1 0 0 00000001 0 0 | 1 0 0 00000003 1 1 | 1 0 0 00000001 | 1 1 0 00000002
T=255000 | 1 | 1 0 1 00000001 0 0 | 1 0 1 00000004 1 1 | 1 0 0 00000001 | 1 0 0 00000003
T=265000 | 1 | 0 0 0 00000000 0 0 | 0 0 0 00000000 1 1 | 1 0 1 00000001 | 1 0 1 00000004
==input port 1 to output port 0,input port 0 to output port 0==========
T=275000 | 1 | 1 1 0 00000001 0 0 | 1 1 0 00000001 0 1 | 0 0 0 00000000 | 0 0 0 00000000
T=285000 | 1 | 1 0 0 00000001 0 0 | 1 1 0 00000002 0 1 | 1 1 0 00000001 | 0 0 0 00000000
T=295000 | 1 | 1 0 0 00000001 0 0 | 1 0 0 00000003 0 1 | 1 0 0 00000001 | 0 0 0 00000000
T=305000 | 1 | 1 0 1 00000001 0 0 | 1 0 1 00000004 0 1 | 1 0 0 00000001 | 0 0 0 00000000
T=315000 | 1 | 0 0 0 00000000 0 0 | 0 0 0 00000000 0 1 | 1 0 1 00000004 | 0 0 0 00000000
==input port 1 to output port 1,input port 0 to output port 1==========
T=325000 | 1 | 1 1 0 00000001 1 0 | 1 1 0 00000001 1 1 | 1 1 0 00000001 | 0 0 0 00000000
T=335000 | 1 | 1 0 0 00000001 1 0 | 1 1 0 00000002 1 1 | 1 0 0 00000001 | 1 1 0 00000001
T=345000 | 1 | 1 0 0 00000001 1 0 | 1 0 0 00000003 1 1 | 1 1 0 00000001 | 1 1 0 00000002
T=355000 | 1 | 1 0 1 00000001 1 0 | 1 0 1 00000004 1 1 | 1 0 0 00000001 | 1 0 0 00000003
T=365000 | 1 | 0 0 0 00000000 1 0 | 1 1 0 00000001 1 1 | 1 1 0 00000001 | 1 0 1 00000004
T=375000 | 1 | 0 0 0 00000000 1 0 | 1 0 1 00000004 1 1 | 1 0 0 00000001 | 1 1 0 00000001
T=385000 | 1 | 0 0 0 00000000 1 0 | 1 0 0 00000004 1 1 | 0 0 0 00000000 | 1 0 1 00000004
T=395000 | 1 | 0 0 0 00000000 1 0 | 1 0 1 00000004 1 1 | 0 0 0 00000000 | 1 0 0 00000004
==========
T=405000 | 1 | 0 0 0 00000000 1 0 | 0 0 0 00000000 1 1 | 0 0 0 00000000 | 1 0 1 00000004
T=415000 | 1 | 0 0 0 00000000 1 0 | 0 0 0 00000000 1 1 | 0 0 0 00000000 | 0 0 0 00000000
T=425000 | 1 | 0 0 0 00000000 1 0 | 0 0 0 00000000 1 1 | 0 0 0 00000000 | 0 0 0 00000000
T=435000 | 1 | 0 0 0 00000000 1 0 | 0 0 0 00000000 1 1 | 0 0 0 00000000 | 0 0 0 00000000
Questa Sim-64 2024.2
RTL
simple_router....
Mate Terminal
QuestaSim
Wave
Transcript

Fifo Issue:



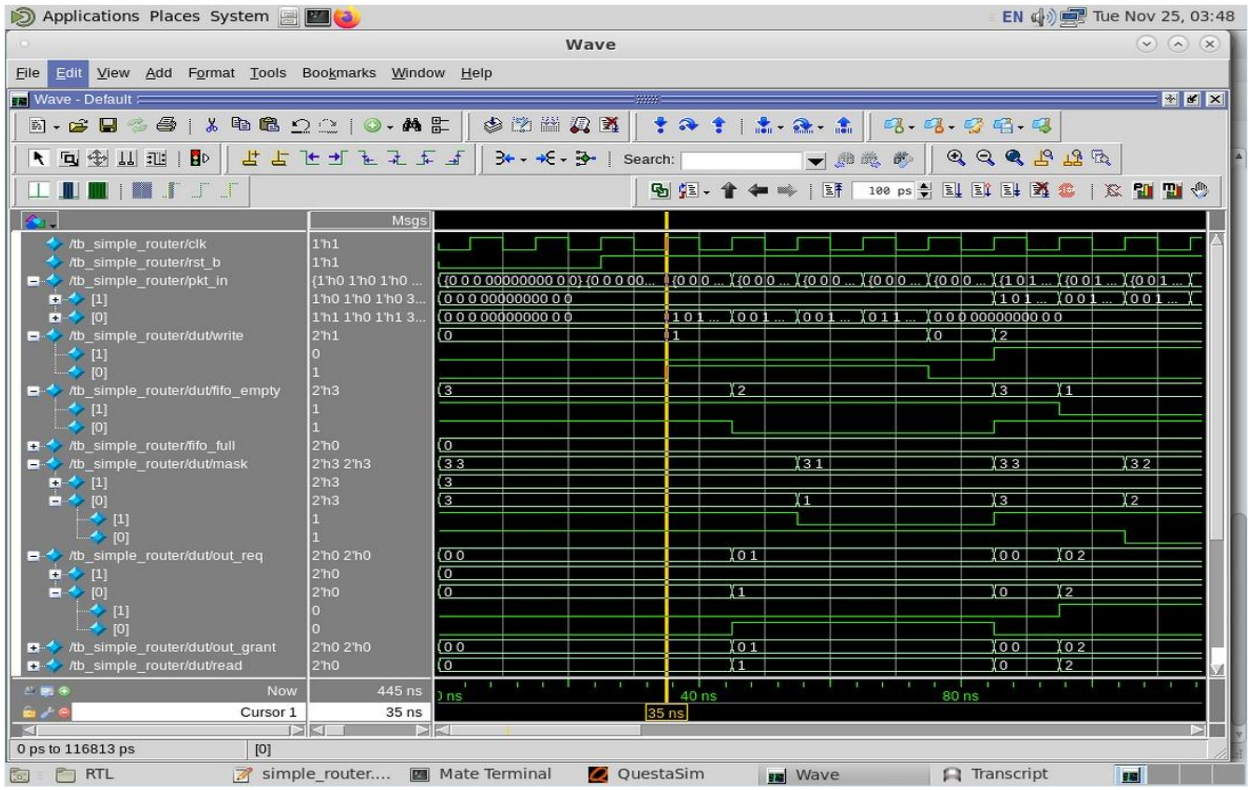
The green-marked area in the screenshot shows that the previously stored head and tail flits from input 0 to FIFO 0 are being sent again through output 0. This happens because the data in FIFO 0 is not being removed after it is read. Since the FIFO cell is not cleared, the same data keeps repeating, and the FIFO is incorrectly seen as non-empty. As a result we are seeing the data flow, the empty signal goes from high to low again and read is high(ouput0 granted to input0), even though no new data is coming from input 0. Because of this, output_0 continues to show valid data, and we observe repeated sequences like head_middle, head_middle, head_middle, even when the actual input is from input 1.

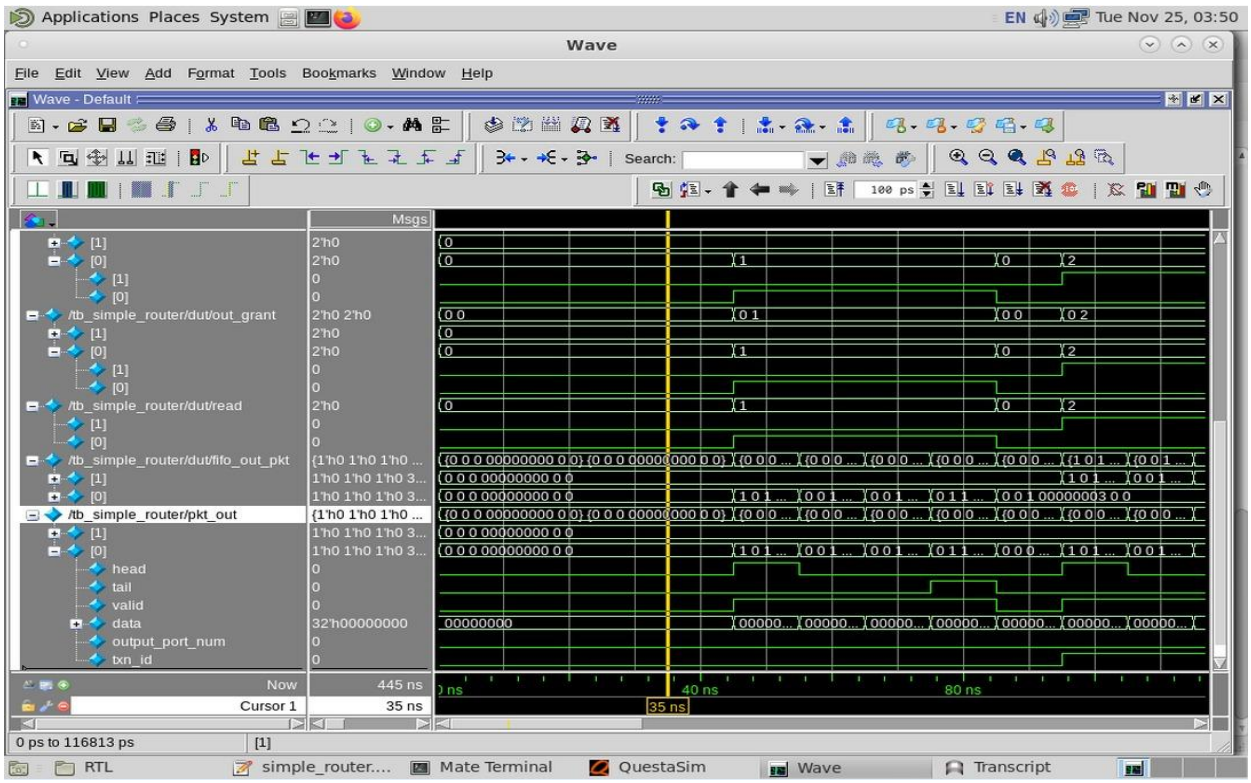
To tackle this issue:

- Increase FIFO depth: This ensures that data from input 0 is not lost due to limited storage. Since the FIFO depth is not specified in the design requirements, this can be considered an improvement suggestion for the design engineer.
- Proper FIFO clearing after read: After reading data from the FIFO, the count should be decremented, and the corresponding FIFO cell should be cleared. This will prevent stale data from being re-read and will avoid generating valid outputs when there is no valid input. It also prevents the masking logic from repeatedly prioritizing the same input and waiting indefinitely for a tail flit.

These changes will help ensure correct data flow and prevent repeated invalid output sequences.

Waveform Analysis:





Transaction from Input[0] => Output[0]

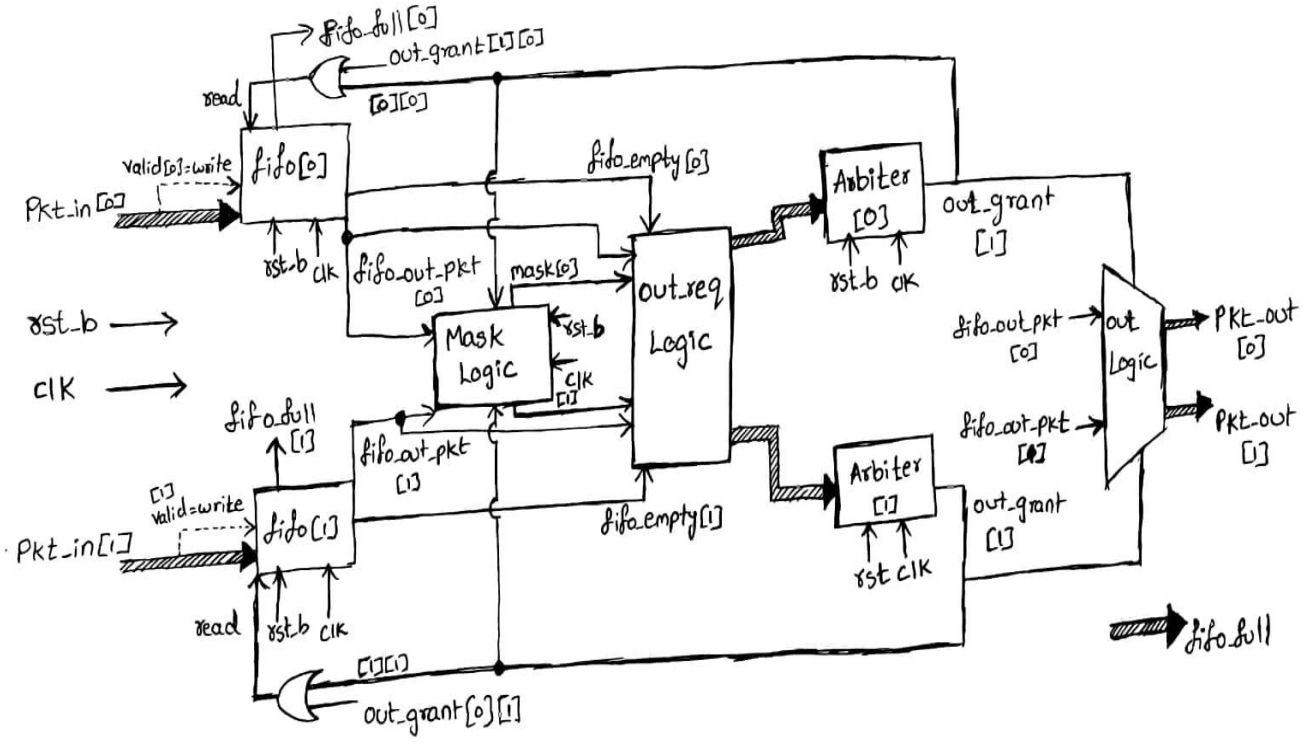
- When the head FLIT is received, the write signal for FIFO[0] goes high and FIFO_EMPTY goes low. This indicates that FIFO[0] now contains data. As a result, the request signal OUT_REQ[0][0] goes high because output_port_num = 0 and mask[0][0] is also high.
- Once OUT_GRANT[0][0] becomes high, the read signal for FIFO[0] is asserted.
- In the same cycle, the head FLIT is read from FIFO[0] and sent to output port 0.
- In a similar manner, the remaining two or three FLITs of the transaction are also read from FIFO[0] and forwarded sequentially to output 0 until the full transaction is completed.

Summary of Dynamic Simulation:

As it is a simple 2x2 router design direct testing was preferred. We just tested the main features of the design and caught most of the bugs here. It is very beneficial to find as many bugs as earlier. If it is a complex design, we use constrained random testing for that. This is the first step of our verification processes, where we understand the given specification (golden rule) and develop a testbench where we check the main feature of the design to start the verification process and gain confidence to taping out.

Part -2 Formal Verification in VC Formal

High-Level Block Diagram



Transaction Description:

A packet enters the router as a sequence of flits through pkt_in[i]. Each valid flit is first stored in the corresponding input FIFO. When a FIFO is not empty, the routing logic checks the output_port_num field and creates a request (out_req) to the required output port, based on the current mask priority.

For each output port, a round-robin arbiter decides which input should be granted access when both inputs request at the same time. This grant signal then triggers a read operation from the selected FIFO and controls a 2-to-1 multiplexer that sends the chosen flit to the correct output port pkt_out[j].

The mask value is updated according to the granted request and the tail flit information to ensure fair access between packets from different inputs.

Timing Diagrams

1. For a transaction going from input 0 to output 0

	A	B	C	D	E	F	G	H
1	Timing diagram describing the basic functionality of a transaction going from input 0 to output 0							
2	clk1	clk2	clk3	clk4	clk5	clk6	clk7	clk8
3	!rst_b	rst_b						
4			pkt_in[0].head pkt_in[0].output_port == 0			pkt_in[0].tail		
5			pkt_in[0].valid, write[0]	pkt_in[0].valid, write[0]	pkt_in[0].valid, write[0]	pkt_in[0].valid, write[0]	! pkt_in[0].valid, ! write[0]	
6			pkt_in[0].data == 32'h1	pkt_in[0].data == 32'h2	pkt_in[0].data == 32'h3	pkt_in[0].data == 32'h4		
7			fifo_empty[0]	! fifo_empty[0]	! fifo_empty[0]	! fifo_empty[0]	! fifo_empty[0]	fifo_empty[0]
8				mask[0] == 11	mask[0] == 01	mask[0] == 01	mask[0] == 01	mask[0] == 11
9				mask[1] == 11	mask[1] == 11	mask[1] == 11	mask[1] == 11	mask[1] == 11
10				req to output_0	req to output_0	req to output_0	req to output_0	
11				gnt to input_0	gnt to input_0	gnt to input_0	gnt to input_0	
12				read to input_0	read to input_0	read to input_0	read to input_0	
13				input_0 fifo_out_0.head	input_0 fifo_out_0.middle	input_0 fifo_out_0.middle	input_0 fifo_out_0.tail	
14				pkt_out[0].head	pkt_out[0].data = 32'h2	pkt_out[0].data = 32'h3	pkt_out[0].tail	

2. For arbiter priority change from input 0 to input 1

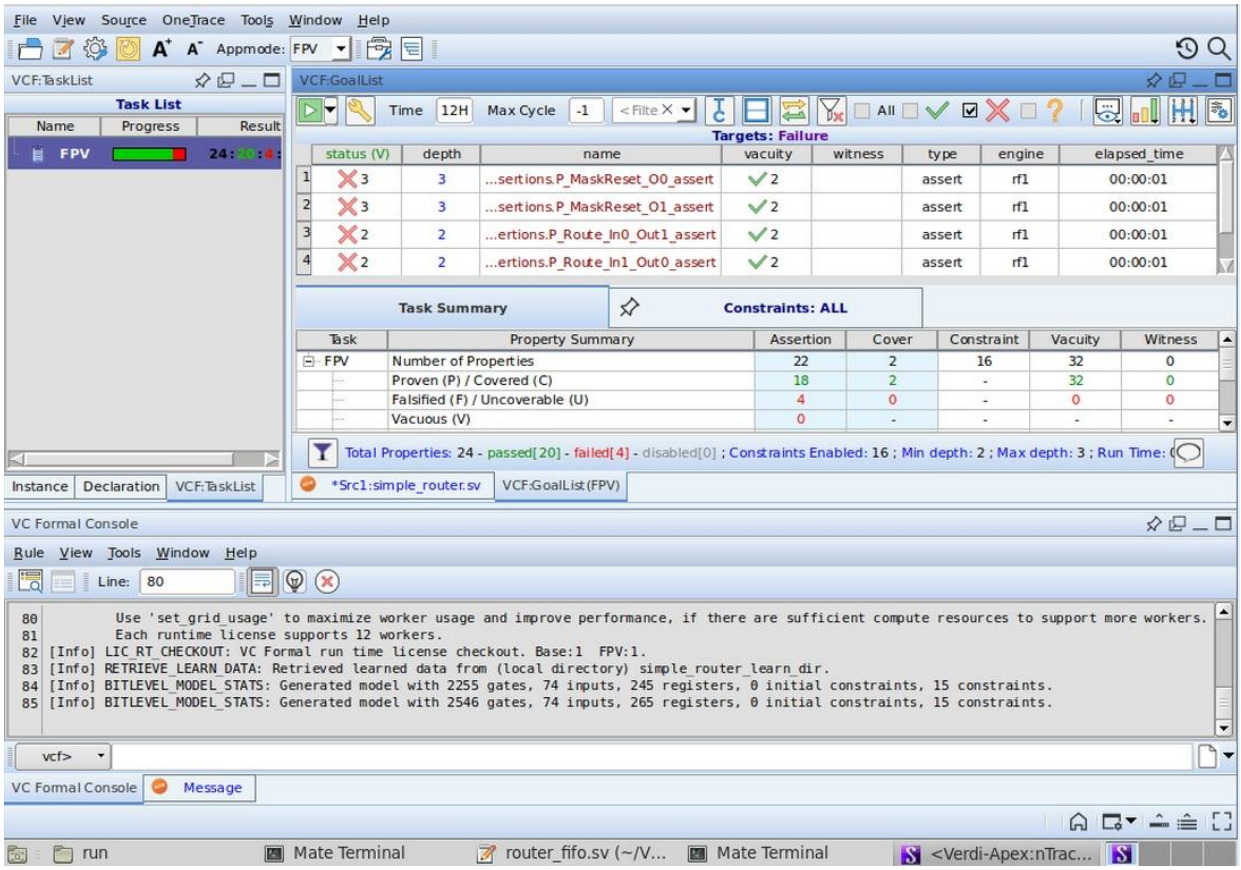
When we send transactions from both inputs for two cycles then we can see the arbiter priority change from input 0 to input 1

	A	B	C	D	E	F	G	H
1							Timing diagram describing the arbiter priority change from input 0	
2	clk1	clk2	clk3	clk4	clk5	clk6	clk7	clk8
3	!rst_b	rst_b						
4			pkt_in[0].head pkt_in[0].output_port == 0 pkt_in[1].head pkt_in[1].out_port == 1			pkt_in[0].tail pkt_in[1].tail		
5			pkt_in[0].valid, write[0] pkt_in[1].valid, write[1]	pkt_in[0].valid, write[0] pkt_in[1].valid, write[1]	pkt_in[0].valid, write[0] pkt_in[1].valid, write[1]	pkt_in[0].valid, write[0] pkt_in[1].valid, write[1]	! pkt_in[0].valid, ! write[0] ! pkt_in[1].valid, ! write[1]	
6			pkt_in[0].data == 32'h1 pkt_in[1].data == 32'h1	pkt_in[0].data == 32'h1 pkt_in[1].data == 32'h2	pkt_in[0].data == 32'h1 pkt_in[1].data == 32'h3	pkt_in[0].data == 32'h1 pkt_in[1].data == 32'h4		
7			fifo_empty[0] fifo_empty[1]	! fifo_empty[0] ! fifo_empty[1]	! fifo_empty[0] ! fifo_empty[1]	! fifo_empty[0] ! fifo_empty[1]	! fifo_empty[0] ! fifo_empty[1]	fifo_empty[0] fifo_empty[1]
8								
9				mask[0] == 11	mask[0] == 01	mask[0] == 01	mask[0] == 01	mask[0] == 11
10				mask[1] == 11	mask[1] == 10	mask[1] == 10	mask[1] == 10	mask[1] == 11
11				inp0_req to output_0 inp1_req to output_1	inp0_req to output_0 inp1_req to output_1	inp0_req to output_0 inp1_req to output_1	inp0_req to output_0 inp1_req to output_1	
12				out0_gnt to input_0 out1_gnt to input_1	out0_gnt to input_0 out1_gnt to input_1	out0_gnt to input_0 out1_gnt to input_1	out0_gnt to input_0 out1_gnt to input_1	
13				read_0 to input_0 read_1 to input_1	read_0 to input_0 read_1 to input_1	read_0 to input_0 read_1 to input_1	read_0 to input_0 read_1 to input_1	
14				input_0 fifo_out_0.head input_1 fifo_out_1.head	input_0 fifo_out_0.middle input_1 fifo_out_1.middle	input_0 fifo_out_0.middle input_1 fifo_out_1.middle	input_0 fifo_out_0.tail input_1 fifo_out_1.tail	
15				pkt_out[0].head pkt_out[1].head	pkt_out[0].data = 32'h1 pkt_out[1].data = 32'h2	pkt_out[0].data = 32'h1 pkt_out[1].data = 32'h3	pkt_out[0].tail pkt_out[1].tail	

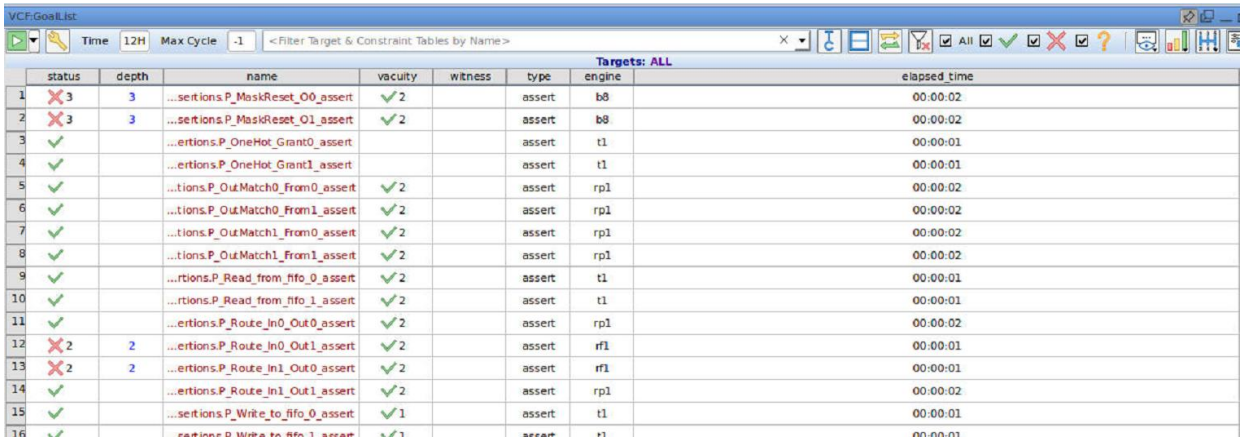
Continuation of the timing diagram in the next page

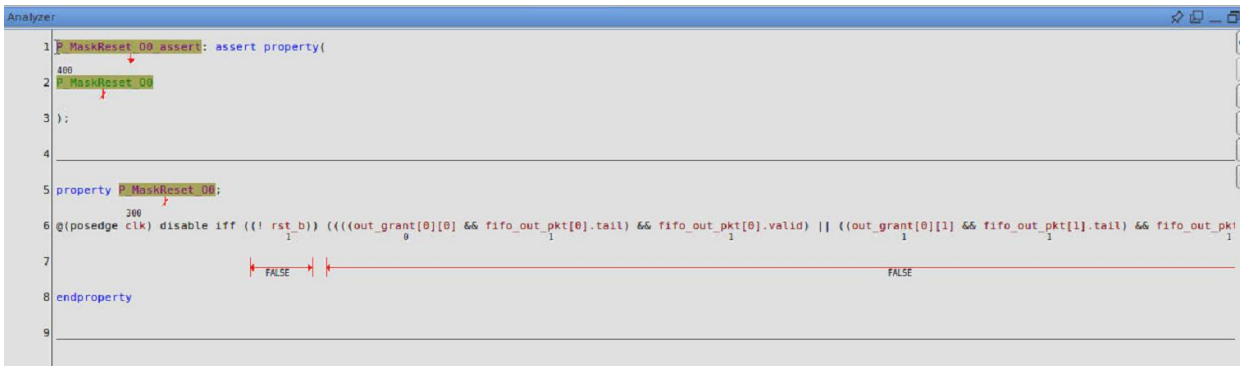
H	I	J	K	L	M	N
change from input 0 to input 1 (giving inputs from both for two transactions)						
clk8	clk9	clk10	clk11	clk12	clk13	clk14
	pkt_in[0].head pkt_in[0].output_port == 0 pkt_in[1].head pkt_in[1].out_port == 0			pkt_in[0].tail pkt_in[1].tail		
	pkt_in[0].valid, write[0] pkt_in[1].valid, write[1]	pkt_in[0].valid, write[0] pkt_in[1].valid, write[1]	pkt_in[0].valid, write[0] pkt_in[1].valid, write[1]	pkt_in[0].valid, write[0] pkt_in[1].valid, write[1]	! pkt_in[0].valid, ! write[0] ! pkt_in[1].valid, ! write[1]	
	pkt_in[0].data == 32'h1 pkt_in[1].data == 32'h1	pkt_in[0].data == 32'h1 pkt_in[1].data == 32'h2	pkt_in[0].data == 32'h1 pkt_in[1].data == 32'h3	pkt_in[0].data == 32'h1 pkt_in[1].data == 32'h4		
fifo_empty[0] fifo_empty[1]	fifo_empty[0] fifo_empty[1]	! fifo_empty[0] ! fifo_empty[1]	! fifo_empty[0] ! fifo_empty[1]	! fifo_empty[0] ! fifo_empty[1]	! fifo_empty[0] ! fifo_empty[1]	! fifo_empty[0] fifo_empty[1]
			fifo_full[0]	fifo_full[0]	fifo_full[0]	fifo_full[0]
mask[0] == 11		mask[0] == 11	mask[0] == 10	mask[0] == 10	mask[0] == 10	mask[0] == 11
mask[1] == 11		mask[1] == 11	mask[1] == 11	mask[1] == 11	mask[1] == 11	mask[1] == 11
		inp0_req to output_0 inp1_req to output_0	inp0_req to output_0 inp1_req to output_0	inp0_req to output_0 inp1_req to output_0	inp0_req to output_0 inp1_req to output_0	
		! out0_gnt to input_0 out0_gnt to input_1	! out0_gnt to input_0 out0_gnt to input_1	! out0_gnt to input_0 out0_gnt to input_1	! out0_gnt to input_0 out0_gnt to input_1	
		! read_0 to input_0 read_1 to input_1	! read_0 to input_0 read_1 to input_1	! read_0 to input_0 read_1 to input_1	! read_0 to input_0 read_1 to input_1	
		input_0 ! fifo_out_0.head input_1 fifo_out_1.head	input_0 ! fifo_out_0.middle input_1 fifo_out_1.middle	input_0 ! fifo_out_0.middle input_1 fifo_out_1.middle	input_0 ! fifo_out_0.tail input_1 fifo_out_1.tail	
		pkt_out[0].head	pkt_out[0].data = 32'h2	pkt_out[0].data = 32'h3	pkt_out[0].tail	

After Compiling VC Formal FPV:

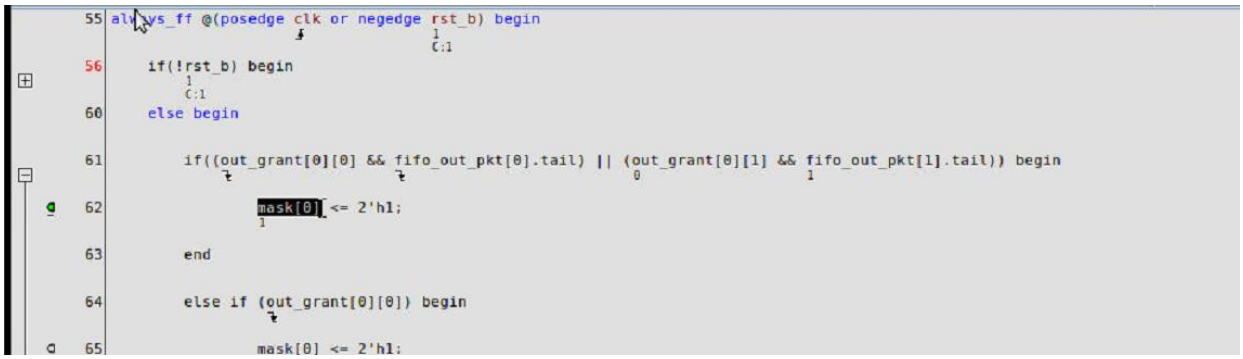


Bug Indentification: After we run the given design

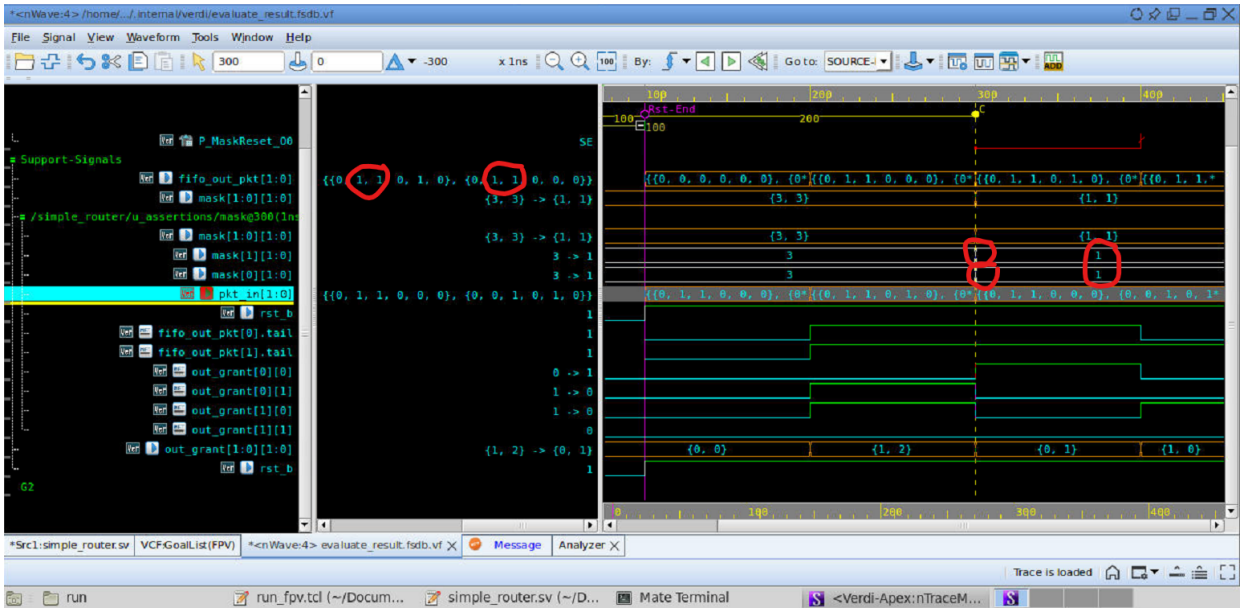




So, MaskReset00 Property has failed. It is checking that after receiving tail flit from either of the inputs to ouput0, mask[0] should be 2'b11.



After clicking driving signals on the mask bits. It took me to the code where we are having issue.



after receiving tail flit from either of the inputs to ouput0 and output1, mask[0] and mask[1] are2'b01. Which is not right. It should be open to both ports. Therefore, we should set mask[0] and mask[1] to 2'b11.

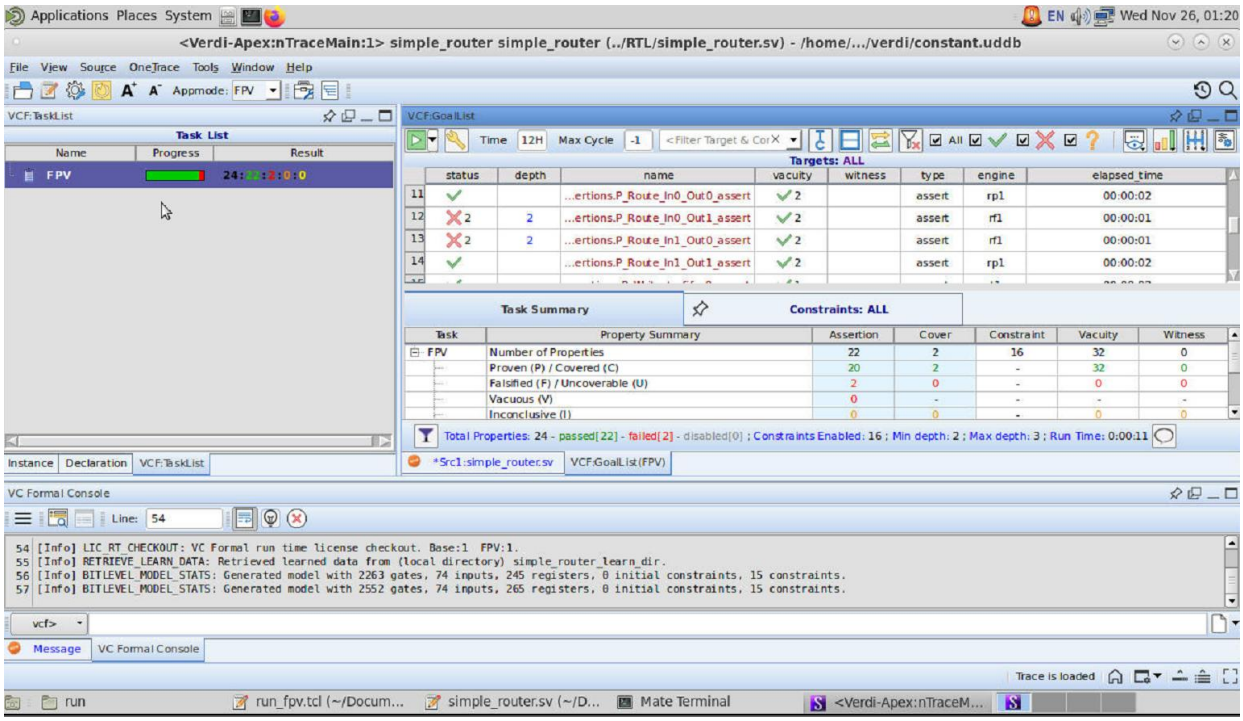
Before Fix:

```
    end
    else begin
        if((out_grant[0][0] && fifo_out_pkt[0].tail) || (out_grant[0][1] && fifo_out_pkt[1].tail)) begin
                mask[0] <= 2'h1;
        end
        else if (out_grant[0][0]) begin
                mask[0] <= 2'h1;
        end
        else if(out_grant[0][1]) begin
                mask[0] <= 2'h2;
        end

        if((out_grant[1][0] && fifo_out_pkt[0].tail) || (out_grant[1][1] && fifo_out_pkt[1].tail)) begin
                mask[1] <= 2'h1;
        end
        else if (out_grant[1][0]) begin
                mask[1] <= 2'h1;
        end
```

After Fix:

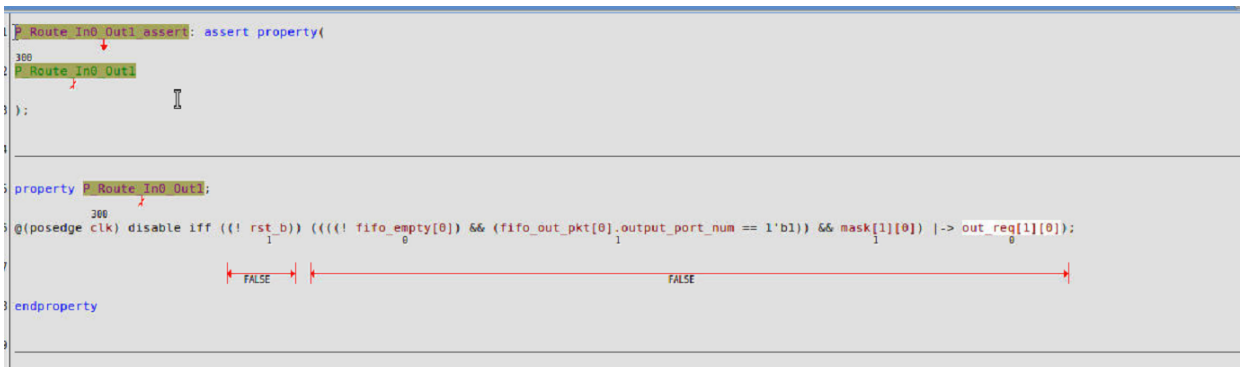
```
        mask[1] <= 2'h1;
    end
    else begin
        if((out_grant[0][0] && fifo_out_pkt[0].tail) || (out_grant[0][1] && fifo_out_pkt[1].tail)) begin
                mask[0] <= 2'h3;
        end
        else if (out_grant[0][0]) begin
                mask[0] <= 2'h1;
        end
        else if(out_grant[0][1]) begin
                mask[0] <= 2'h2;
        end

        if((out_grant[1][0] && fifo_out_pkt[0].tail) || (out_grant[1][1] && fifo_out_pkt[1].tail)) begin
                mask[1] <= 2'h3;
        end
```

After running it again:

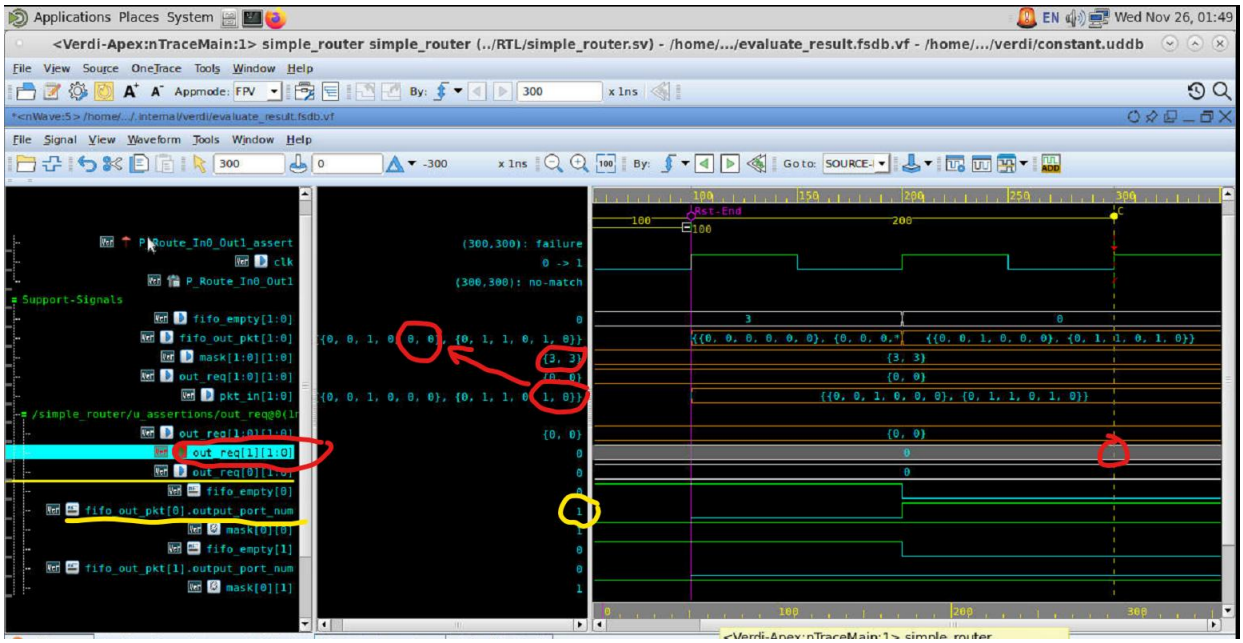


There are 2 remaining failed assertions.

After double clicking on the x mark for failed assertion P_Route_In0_Out1. I found out that out_req[1][0] is low where it should be high. Let's check the wave form to check why it is 0.



After seeing waveform. I noticed that flit from input 0 is not going to output 0 as out_req[1][0] is zero.



Even though the output port num of input 0 flit is 1, Out_req[1][0] is low. Now we check out_req[1][0] trace to find bug.

Signals/Drivers/Loads	Value	Time	File(Line)	
out_req	{0, 0}	0	../RTL/route...ions.sva(11)	simple_router.u_assertions
assign out_req[0][0] = !fi...= 0) && mask[0][0] : 1'b0;		0	../RTL/simpl...outer.sv(40)	simple_router
assign out_req[0][1] = !fi...= 1) && mask[0][1] : 1'b0;		0	../RTL/simpl...outer.sv(41)	simple_router
assign out_req[1][0] = !fi...= 0) && mask[1][0] : 1'b0;		0	../RTL/simpl...outer.sv(42)	simple_router
assign out_req[1][1] = !fi...= 1) && mask[1][1] : 1'b0;		0	../RTL/simpl...outer.sv(43)	simple_router
fifo_out_pkt[0].output_port_num	NF	0	../RTL/simpl...outer.sv(18)	simple_router
assign fifo_out_pkt = fifo_mem[rd_ptr];		0	../RTL/router_fifo.sv(53)	simple_router.router_inp_fifo[0].i_router_fifo
fifo_out_pkt[1].output_port_num	NF	0	../RTL/simpl...outer.sv(18)	simple_router

```
40 assign out_req[0][0] = !fifo_empty[0] ? (fifo_out_pkt[0].output_port_num == 0) && mask[0][0] : 1'b0;
41 assign out_req[0][1] = !fifo_empty[1] ? (fifo_out_pkt[1].output_port_num == 1) && mask[0][1] : 1'b0;
42 assign out_req[1][0] = !fifo_empty[0] ? (fifo_out_pkt[0].output_port_num == 0) && mask[1][0] : 1'b0;
43 assign out_req[1][1] = !fifo_empty[1] ? (fifo_out_pkt[1].output_port_num == 1) && mask[1][1] : 1'b0;
44
```

After carefully observing the out_req[1][0] logic. It is comparing fifo_out_pkt[0].output_port_num to the 0 which should be 1. This is the bug right here in the code.

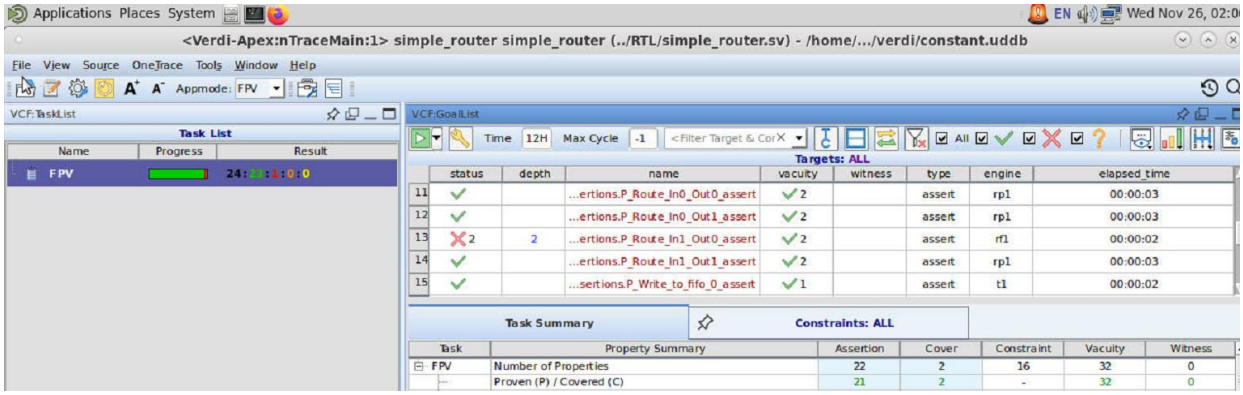
Before fixing the code:

```
endgenerate

assign out_req[0][0] = !fifo_empty[0] ? (fifo_out_pkt[0].output_port_num == 0) && mask[0][0] : 1'b0;
assign out_req[0][1] = !fifo_empty[1] ? (fifo_out_pkt[1].output_port_num == 1) && mask[0][1] : 1'b0;
assign out_req[1][0] = !fifo_empty[0] ? (fifo_out_pkt[0].output_port_num == 0) && mask[1][0] : 1'b0;
assign out_req[1][1] = !fifo_empty[1] ? (fifo_out_pkt[1].output_port_num == 1) && mask[1][1] : 1'b0;
```

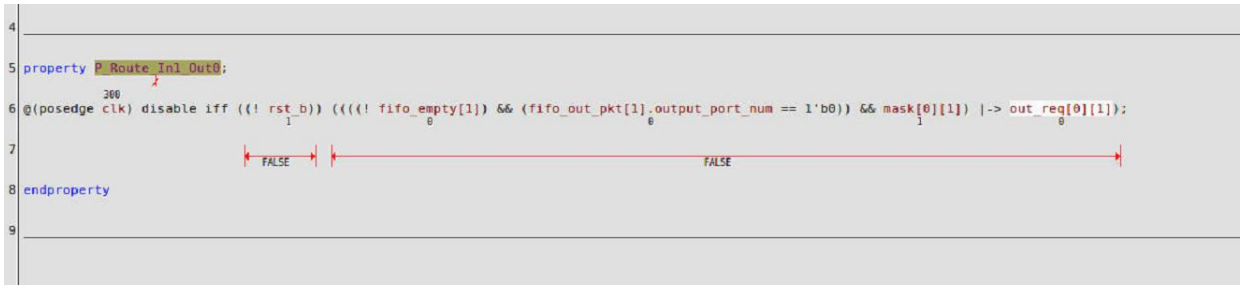
After fixing the code:

```
assign out_req[0][0] = !fifo_empty[0] ? (fifo_out_pkt[0].output_port_num == 0) && mask[0][0] : 1'b0;
assign out_req[0][1] = !fifo_empty[1] ? (fifo_out_pkt[1].output_port_num == 1) && mask[0][1] : 1'b0;
assign out_req[1][0] = !fifo_empty[0] ? (fifo_out_pkt[0].output_port_num == 1) && mask[1][0] : 1'b0;
assign out_req[1][1] = !fifo_empty[1] ? (fifo_out_pkt[1].output_port_num == 1) && mask[1][1] : 1'b0;
```

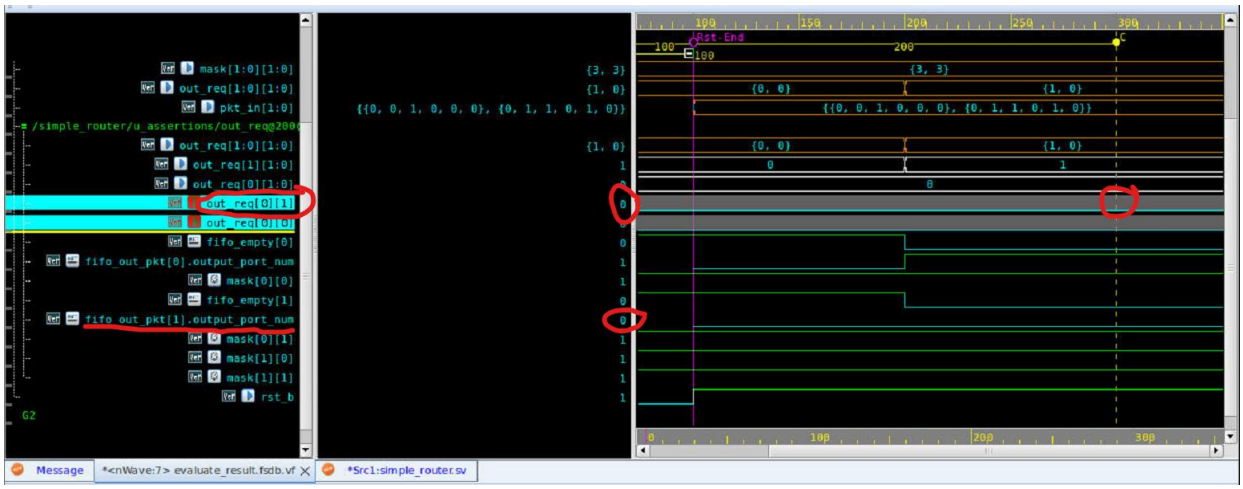
After running it again:



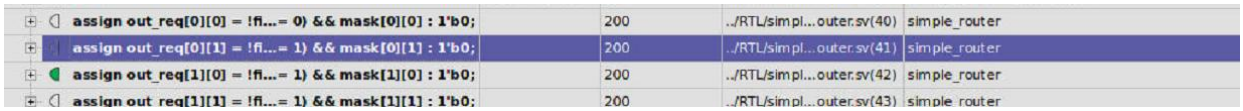
We still have 1 failed assertion. Lets check the CEX of it.

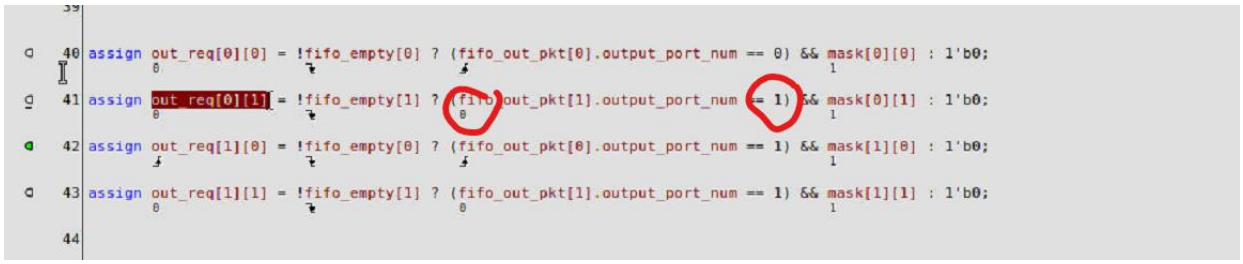


The out_req[0][1] is 0. This was similar to the above bug in the out_req[1][0] logic. Lets check waveform.



As suspected, we encountered that there is an issue with out_req[0][1] logic. Let's check trace





Fifo_out_pkt[1].output_port_num should be compare to output0 (value=0) but it is comparing to the value 1. This is the bug in the out_req[0][1] logic.

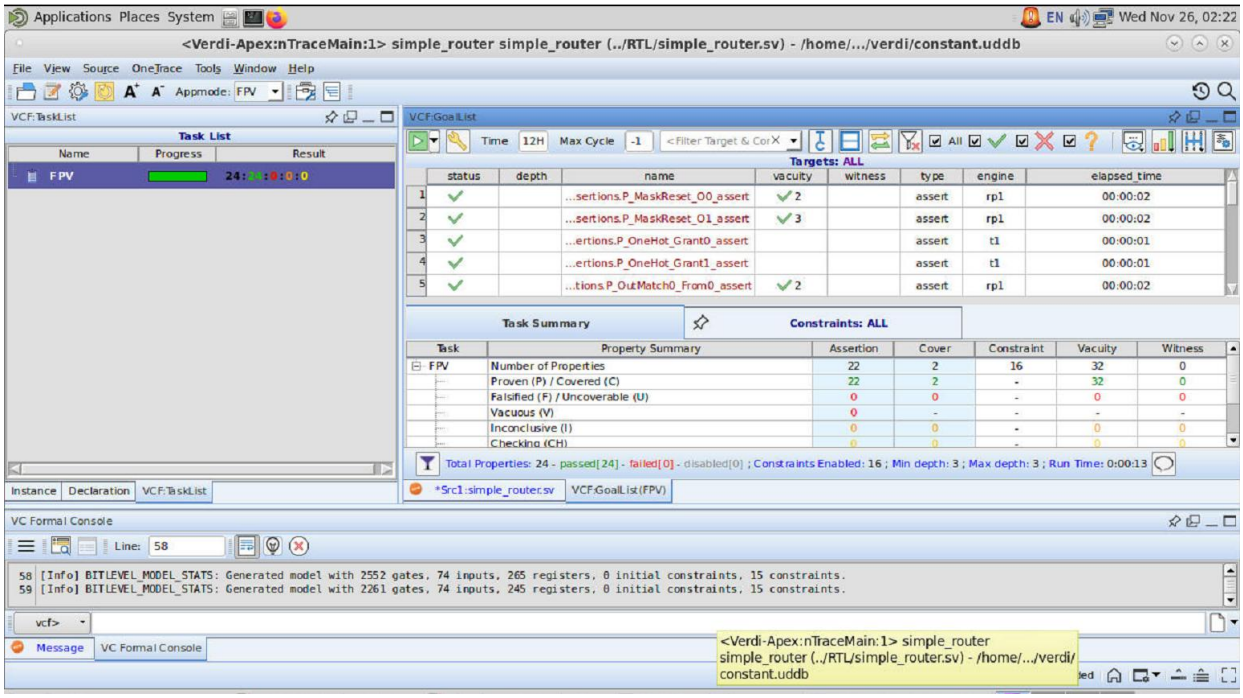
Before Fix:

```
assign out_req[0][0] = !fifo_empty[0] ? (fifo_out_pkt[0].output_port_num == 0) && mask[0][0] : 1'b0;
assign out_req[0][1] = !fifo_empty[1] ? (fifo_out_pkt[1].output_port_num == 1) && mask[0][1] : 1'b0;
assign out_req[1][0] = !fifo_empty[0] ? (fifo_out_pkt[0].output_port_num == 1) && mask[1][0] : 1'b0;
assign out_req[1][1] = !fifo_empty[1] ? (fifo_out_pkt[1].output_port_num == 1) && mask[1][1] : 1'b0;
```

After Fix:

```
endgenerate

assign out_req[0][0] = !fifo_empty[0] ? (fifo_out_pkt[0].output_port_num == 0) && mask[0][0] : 1'b0;
assign out_req[0][1] = !fifo_empty[1] ? (fifo_out_pkt[1].output_port_num == 0) && mask[0][1] : 1'b0;
assign out_req[1][0] = !fifo_empty[0] ? (fifo_out_pkt[0].output_port_num == 1) && mask[1][0] : 1'b0;
assign out_req[1][1] = !fifo_empty[1] ? (fifo_out_pkt[1].output_port_num == 1) && mask[1][1] : 1'b0;
```

FPV Simulation Output after Bug Fix:

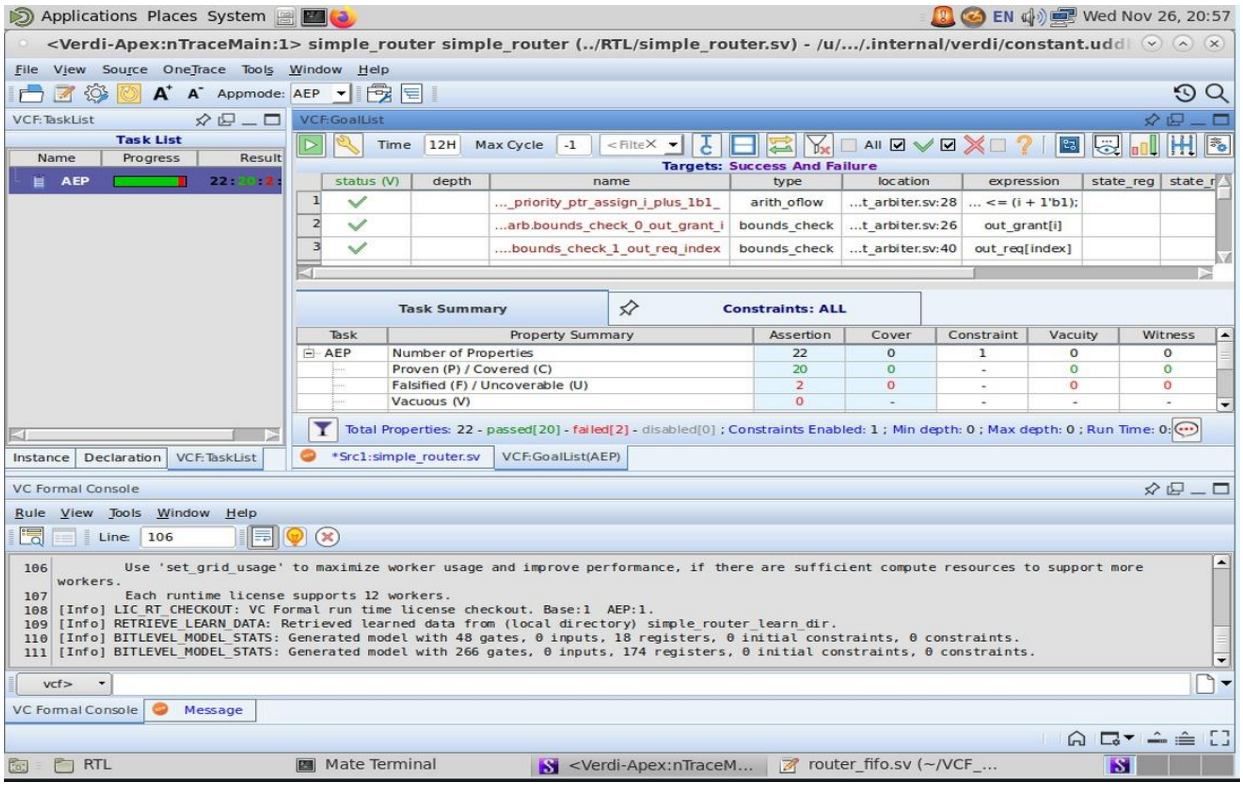


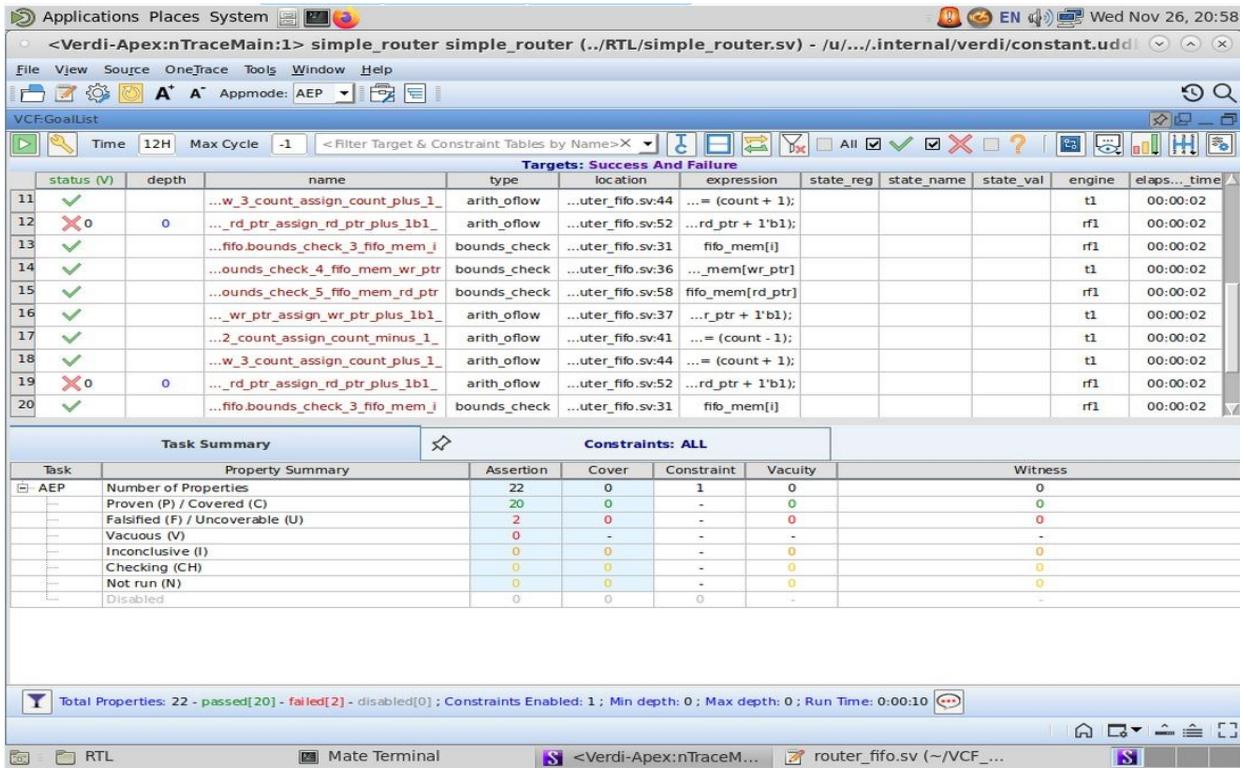
Show usage of different app other than FPV. Explain what app did you use? Why you chose that and how did it help your verification.

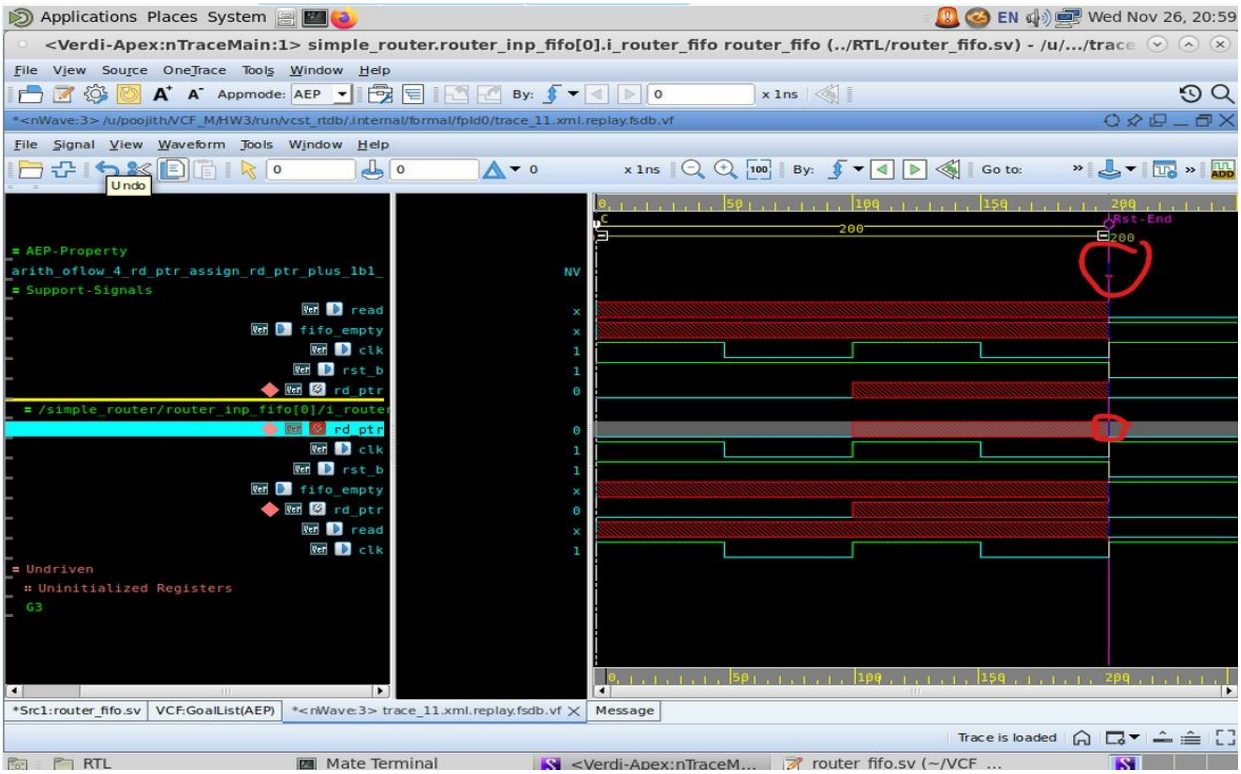
AEP (Automatic Extracted Properties):

I choose AEP because while I was running FPV in Jaspergold, I got a combinational loop error, which I didn't get in FPV of VC Formal. So, to check any other failed mathematically properties in VC Formal that will be failed, I choose AEP.

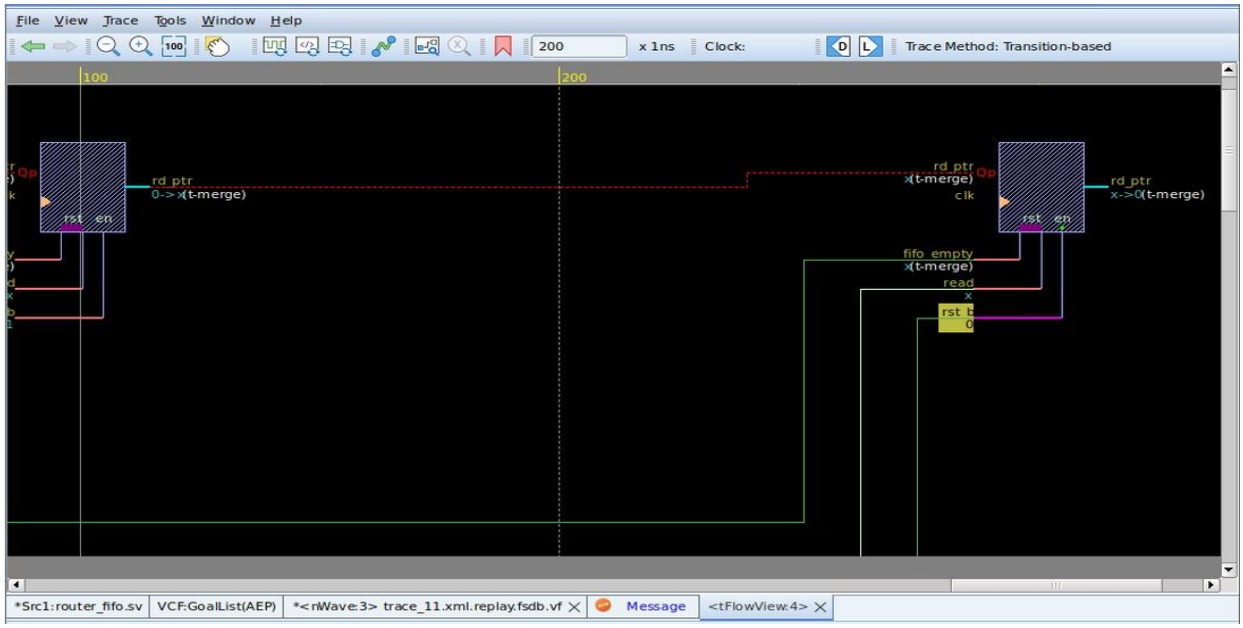
AEP (Automatic Exhaustive Proof) in VC Formal is used to automatically analyze the design for structural and control issues such as deadlocks, unreachable states, and FSM completeness without requiring user-defined assertions. It helps identify hidden design flaws by exhaustively exploring all possible state transitions and control paths.


Applications Places System
Wed Nov 26, 20:57
<Verdi-Apex:nTraceMain:1> simple_router simple_router (../RTL/simple_router.sv) - /u/.../.internal/verdi/constant.udd
File View Source OneTrace Tools Window Help
Appmode: AEP
VCF:TaskList
Task List
Name Progress Result
AEP
VCF:GoalList
Time 12H Max Cycle -1
Targets: Success And Failure
status (V) depth name type location expression state_reg state_r
1 ..._priority_ptr_assign_i_plus_1b1_ arith_oflow ...t_arbiter.sv:28 ... <= (i + 1'b1);
2 ...arb.bounds_check_0_out_grant_i bounds_check ...t_arbiter.sv:26 out_grant[i]
3 ....bounds_check_1_out_req_index bounds_check ...t_arbiter.sv:40 out_req[index]
Task Summary
Constraints: ALL
Task Property Summary Assertion Cover Constraint Vacuity Witness
AEP Number of Properties 22 0 1 0 0
Proven (P) / Covered (C) 20 0 - 0 0
Falsified (F) / Uncoverable (U) 2 0 - 0 0
Vacuous (V) 0 - - - -
Total Properties: 22 - passed[20] - failed[2] - disabled[0] ; Constraints Enabled: 1 ; Min depth: 0 ; Max depth: 0 ; Run Time: 0:
Instance Declaration VCF:TaskList
*Src1:simple_router.sv VCF:GoalList(AEP)
VC Formal Console
Rule View Tools Window Help
Line: 106
106 Use 'set_grid_usage' to maximize worker usage and improve performance, if there are sufficient compute resources to support more workers.
107 Each runtime license supports 12 workers.
108 [Info] LIC_RT_CHECKOUT: VC Formal run time license checkout. Base:1 AEP:1.
109 [Info] RETRIEVE_LEARN_DATA: Retrieved learned data from (local directory) simple_router_learn_dir.
110 [Info] BITLEVEL_MODEL_STATS: Generated model with 48 gates, 0 inputs, 18 registers, 0 initial constraints, 0 constraints.
111 [Info] BITLEVEL_MODEL_STATS: Generated model with 266 gates, 0 inputs, 174 registers, 0 initial constraints, 0 constraints.
vcf>
VC Formal Console Message
RTL
Mate Terminal
<Verdi-Apex:nTraceM...
router_fifo.sv (~/VCF_...


Applications Places System
Wed Nov 26, 20:58
<Verdi-Apex:nTraceMain:1> simple_router simple_router (../RTL/simple_router.sv) - /u/.../.internal/verdi/constant.udd
File View Source OneTrace Tools Window Help
Appmode: AEP
VCF:GoalList
Time 12H Max Cycle -1
Targets: Success And Failure
status (V) depth name type location expression state_reg state_name state_val engine elaps..._time
11 ...w_3_count_assign_count_plus_1_ arith_oflow ...uter_fifo.sv:44 ...= (count + 1); t1 00:00:02
12 0 0 ..._rd_ptr_assign_rd_ptr_plus_1b1_ arith_oflow ...uter_fifo.sv:52 ...rd_ptr + 1'b1); rf1 00:00:02
13 ...fifo.bounds_check_3_fifo_mem_i bounds_check ...uter_fifo.sv:31 fifo_mem[i] rf1 00:00:02
14 ...ounds_check_4_fifo_mem_wr_ptr bounds_check ...uter_fifo.sv:36 ..._mem[wr_ptr] t1 00:00:02
15 ...ounds_check_5_fifo_mem_rd_ptr bounds_check ...uter_fifo.sv:58 fifo_mem[rd_ptr] rf1 00:00:02
16 ..._wr_ptr_assign_wr_ptr_plus_1b1_ arith_oflow ...uter_fifo.sv:37 ...r_ptr + 1'b1); t1 00:00:02
17 ...2_count_assign_count_minus_1_ arith_oflow ...uter_fifo.sv:41 ...= (count - 1); t1 00:00:02
18 ...w_3_count_assign_count_plus_1_ arith_oflow ...uter_fifo.sv:44 ...= (count + 1); t1 00:00:02
19 0 0 ..._rd_ptr_assign_rd_ptr_plus_1b1_ arith_oflow ...uter_fifo.sv:52 ...rd_ptr + 1'b1); rf1 00:00:02
20 ...fifo.bounds_check_3_fifo_mem_i bounds_check ...uter_fifo.sv:31 fifo_mem[i] rf1 00:00:02
Task Summary
Constraints: ALL
Task Property Summary Assertion Cover Constraint Vacuity Witness
AEP Number of Properties 22 0 1 0 0
Proven (P) / Covered (C) 20 0 - 0 0
Falsified (F) / Uncoverable (U) 2 0 - 0 0
Vacuous (V) 0 - - - -
Inconclusive (I) 0 0 - 0 0
Checking (CH) 0 0 - 0 0
Not run (N) 0 0 - 0 0
Disabled 0 0 0 -
Total Properties: 22 - passed[20] - failed[2] - disabled[0] ; Constraints Enabled: 1 ; Min depth: 0 ; Max depth: 0 ; Run Time: 0:00:10
RTL
Mate Terminal
<Verdi-Apex:nTraceM...
router_fifo.sv (~/VCF_...



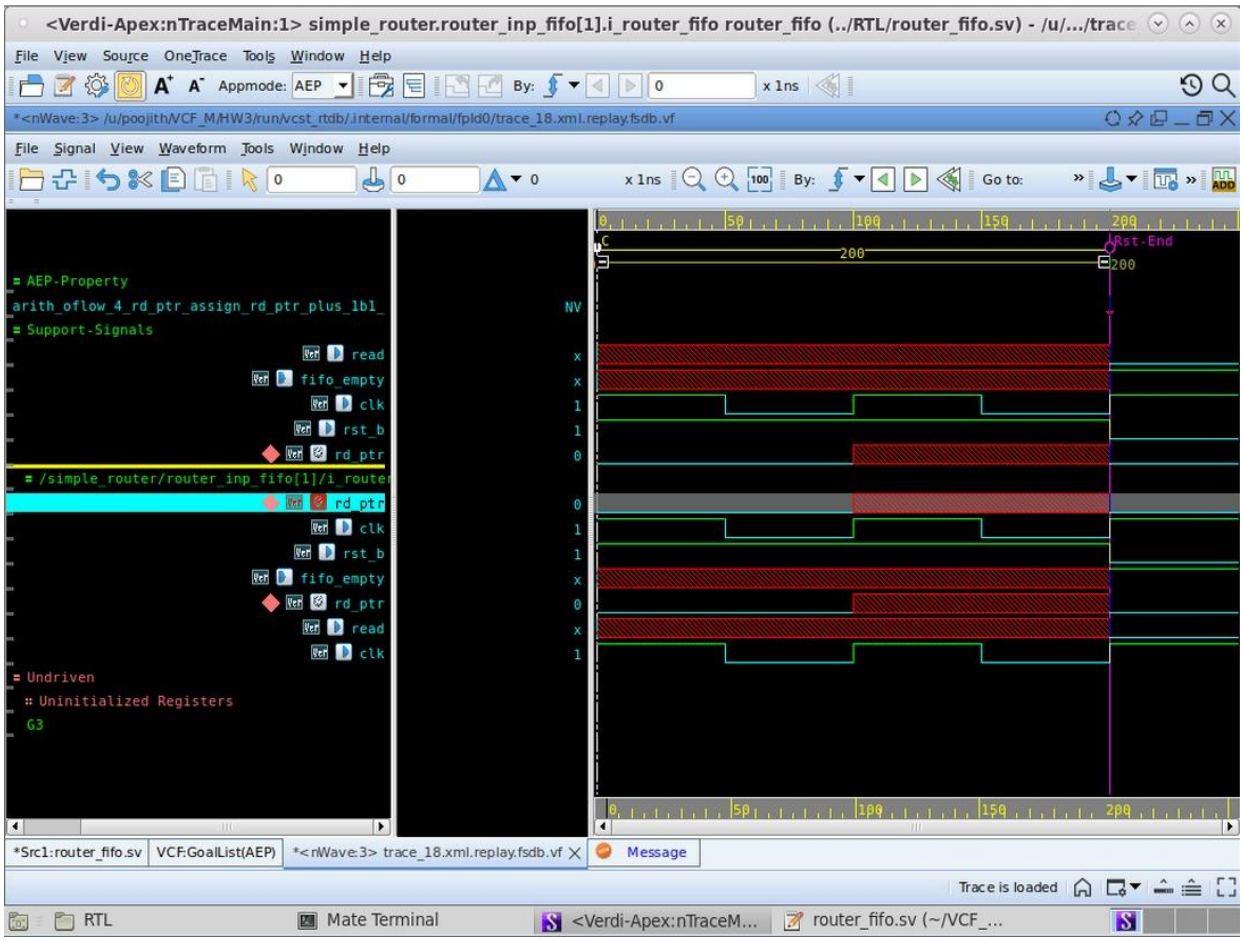
The arithmetic overflow property of read pointer is failing because it is going out of bound with least guarding signals and high controllability to AEP app. So, this is the reason we are not getting failed arithmetic overflow assertions for write pointer, as it has more guarding signals and less controllability compared to AEP compared to read pointer. So, failing to write takes much more complex sequence or a particular state of guarding signals.



```
25    // Asynchronous reset
26    if(~rst_b) begin
27       wr_ptr      <= '0;
28       rd_ptr      <= '0;
29       count <= '0;
30       for(int i = 0; i < DEPTH; i++) begin
31          fifo_mem[i] <= '0;
32       end
```

```
*Src1:simple_router.router_inp_fifo[0].i_router_fifo(/u/poojith/VCF_M/HW3/RTL/router_fifo.sv)
50
51  if(read && !fifo_empty) begin
52        rd_ptr <= rd_ptr + 1'b1;
53    end
54  end
55
```

2nd failed AEP CEX:



```
46  /*     if(read && !fifo_empty) begin
47          rd_ptr <= rd_ptr + 1'b1;
48        end */
49     end
50
51  if(read && !fifo_empty) begin
52          rd_ptr <= rd_ptr + 1'b1;
53     end
54  end
55
```

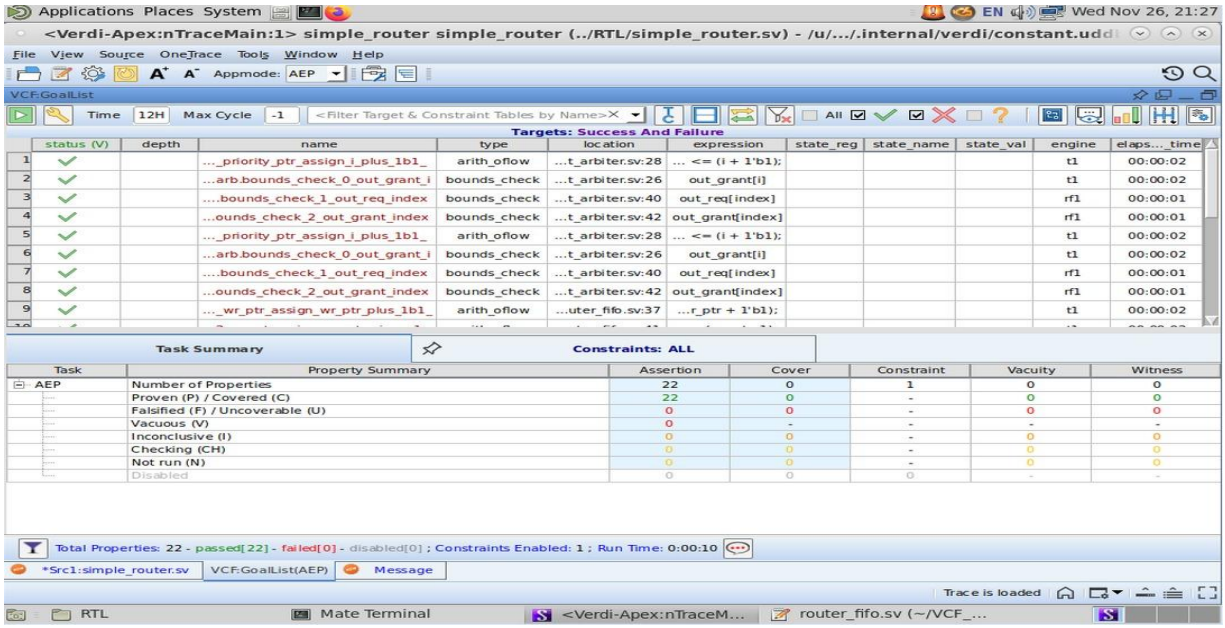
The arithmetic overflow property of read pointer is failing because it is going out of bound with least guarding signals and high controllability to AEP app. So, this is the reason we are not getting failed arithmetic overflow assertions for write pointer, as it has more guarding signals and less controllability compared to AEP compared to read pointer. So, failing to write takes much more complex sequence or a particular state of guarding signals.

This error is not same as combinational loop. In jasper gold we encountered fifo empty was the signal causing the loop and here in AEP the reset is causing the failure of arithmetic overflow assertion. And this is the final debugging of read pointer increment when fifo empty is low and reset is low.

After Improvement:

```
// Synchronous write and pointer update behavior
always_ff @(posedge clk or negedge rst_b)
begin
   // Asynchronous reset
   if(~rst_b) begin
      wr_ptr     <= '0;
      rd_ptr     <= '0;
      count <= '0;
      for(int i = 0; i < DEPTH; i++) begin
         fifo_mem[i] <= '0;
      end
   end
   else begin
   if(write && !fifo_full) begin
         fifo_mem[wr_ptr] <= fifo_in_pkt;  // If we receive a write command, store the input packet to the slot pointed to
by write pointer
         wr_ptr <= wr_ptr + 1'b1;          // Also increment the write pointer
   end

   if(read && !fifo_empty && !write) begin
      count <= count - 1;
   end
   else if(write && !fifo_full && !read) begin
      count <= count + 1;
   end
     if(read && !fifo_empty) begin     //Improvement After Jasper Gold FPV app and VC Formal AEP app.
    rd_ptr <= rd_ptr + 1'b1;
      end
   end


end
```

Result After Improvement:

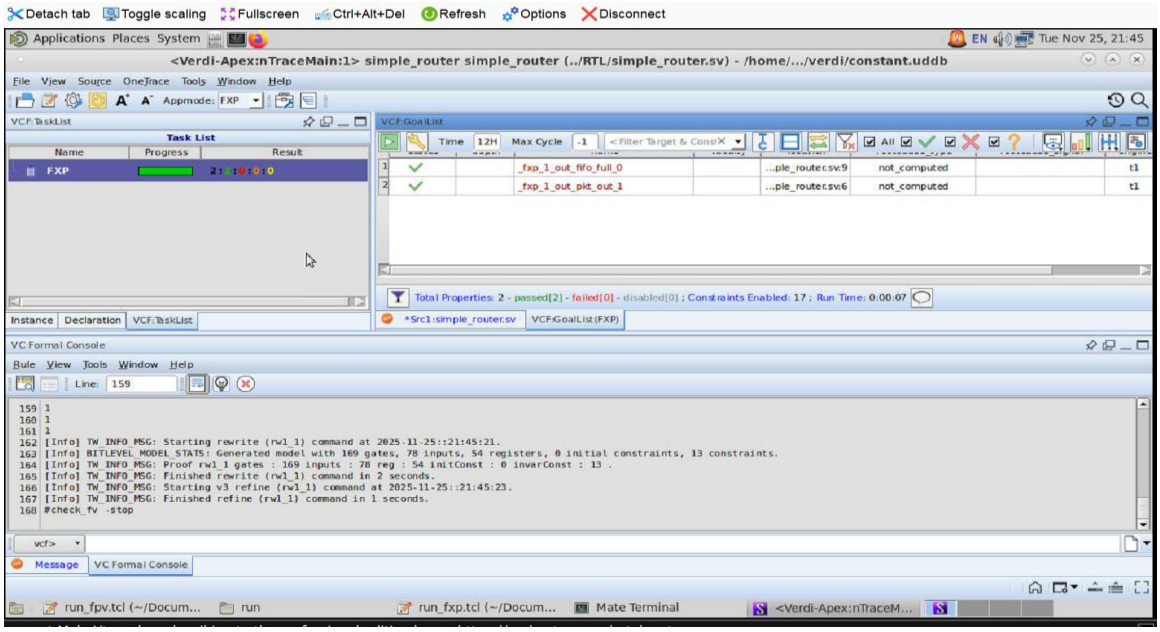


FXP(Formal X Propagation Verification) App:

I choose to use the FXP (Formal X-Propagation) application in VC Formal. The goal was to analyze how unknown values (X-states) propagate through the router logic, especially during reset, masking, FIFO operation, and arbitration.

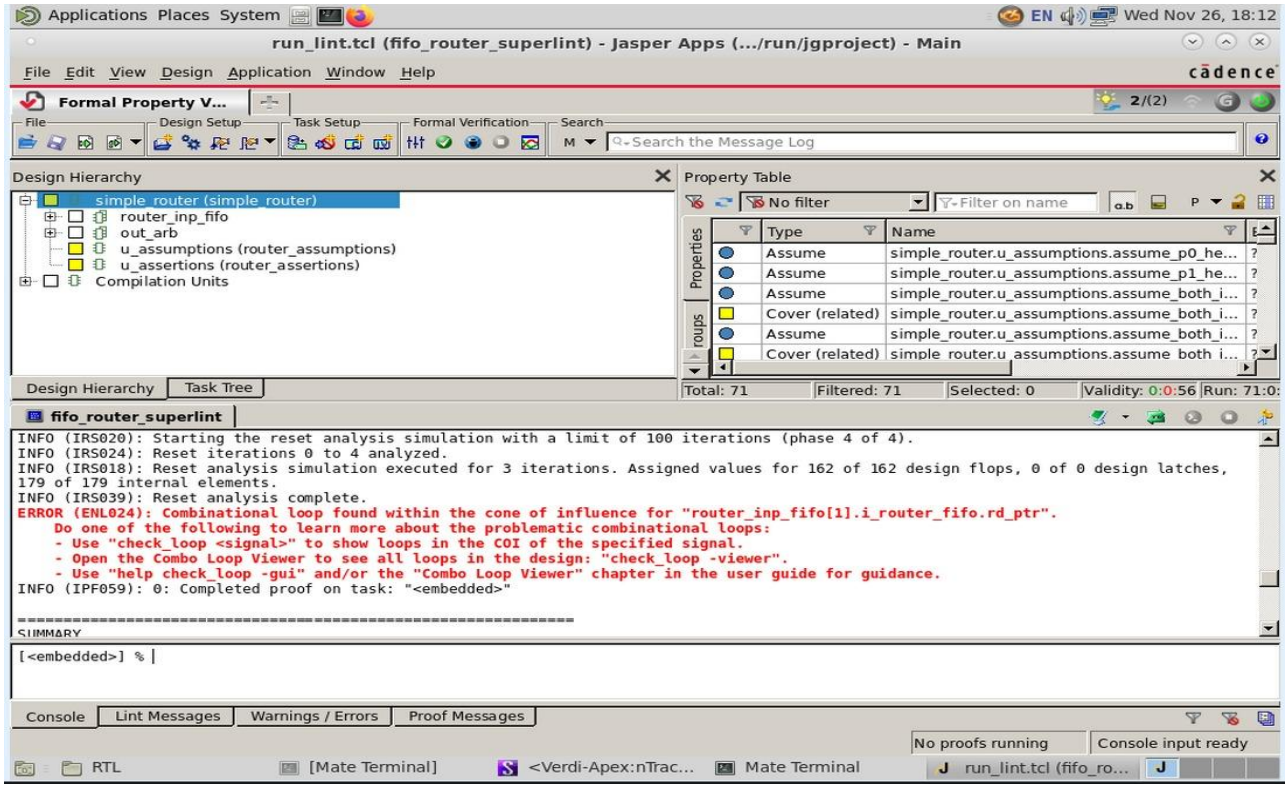
I have chosen to use FXP because the simple_router design has several areas where X-propagation is very likely:

FXP Simulation Output(VC Formal):



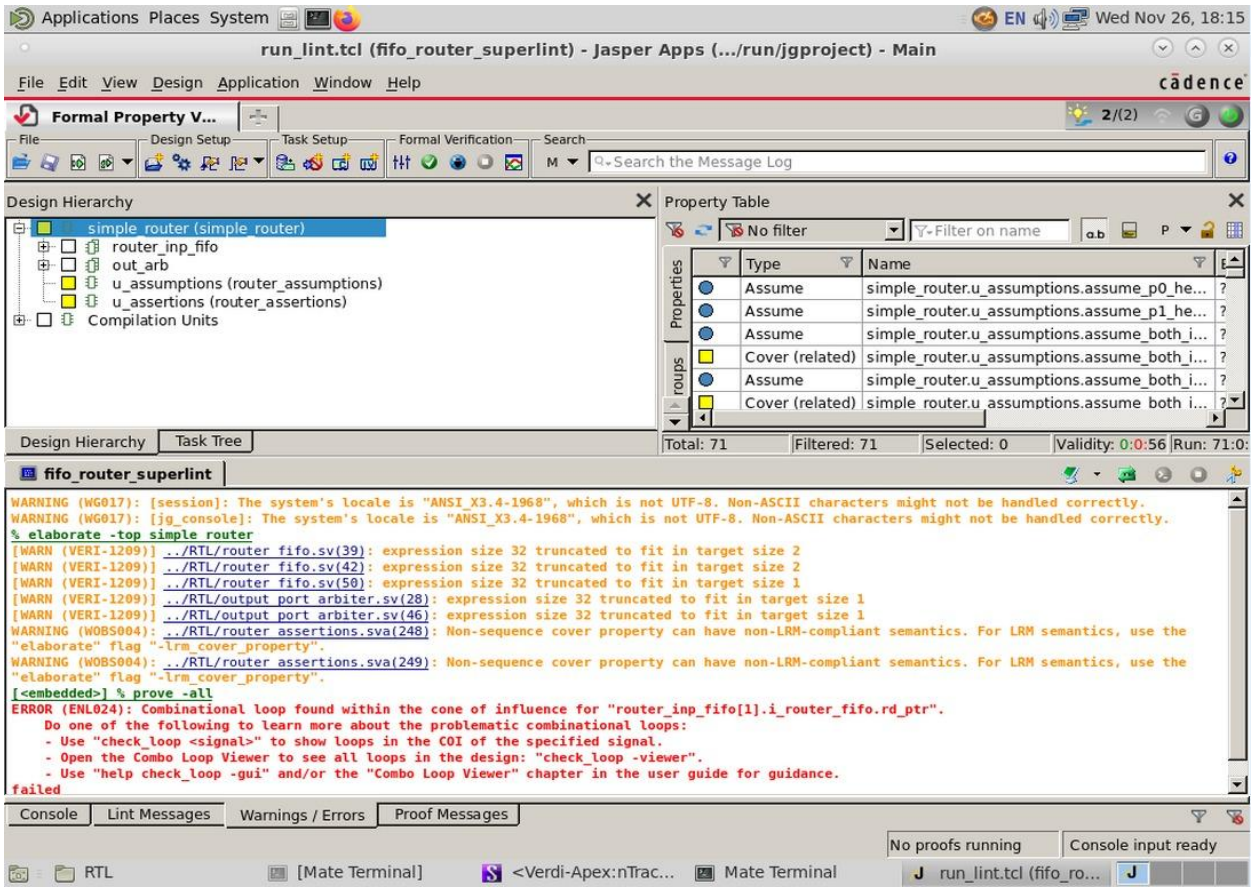
Part -3 Formal Verification in Jasper Gold.

After running the given design



To check the combinational loop, run the command in the console “check_loop -viewer”

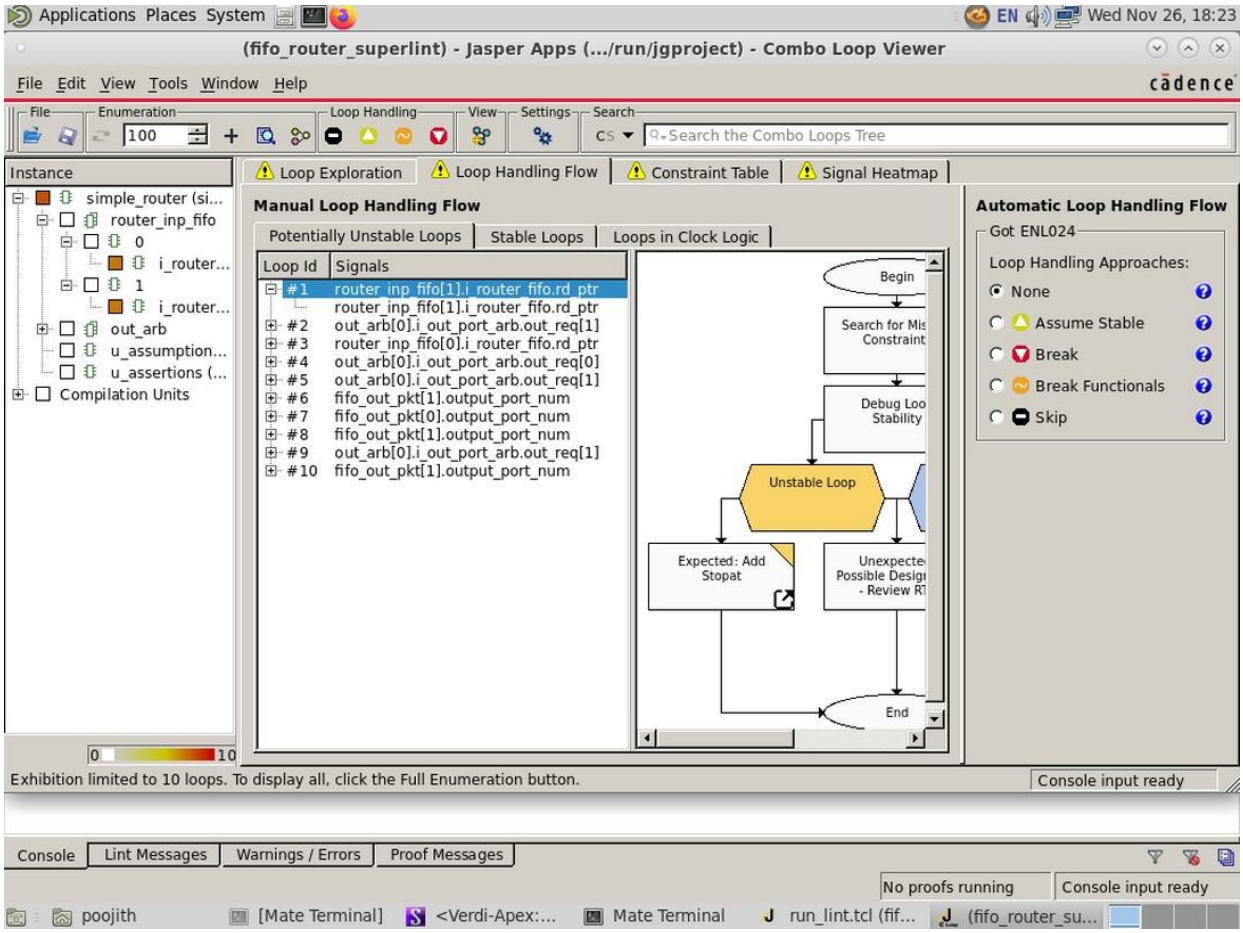
This error was detected in Jasper gold and not in VC Formal, therefore Jasper Gold is slightly better as it detects combinational loops. In VC Formal we did run it through AEP, but we cannot find it in the AEP app.

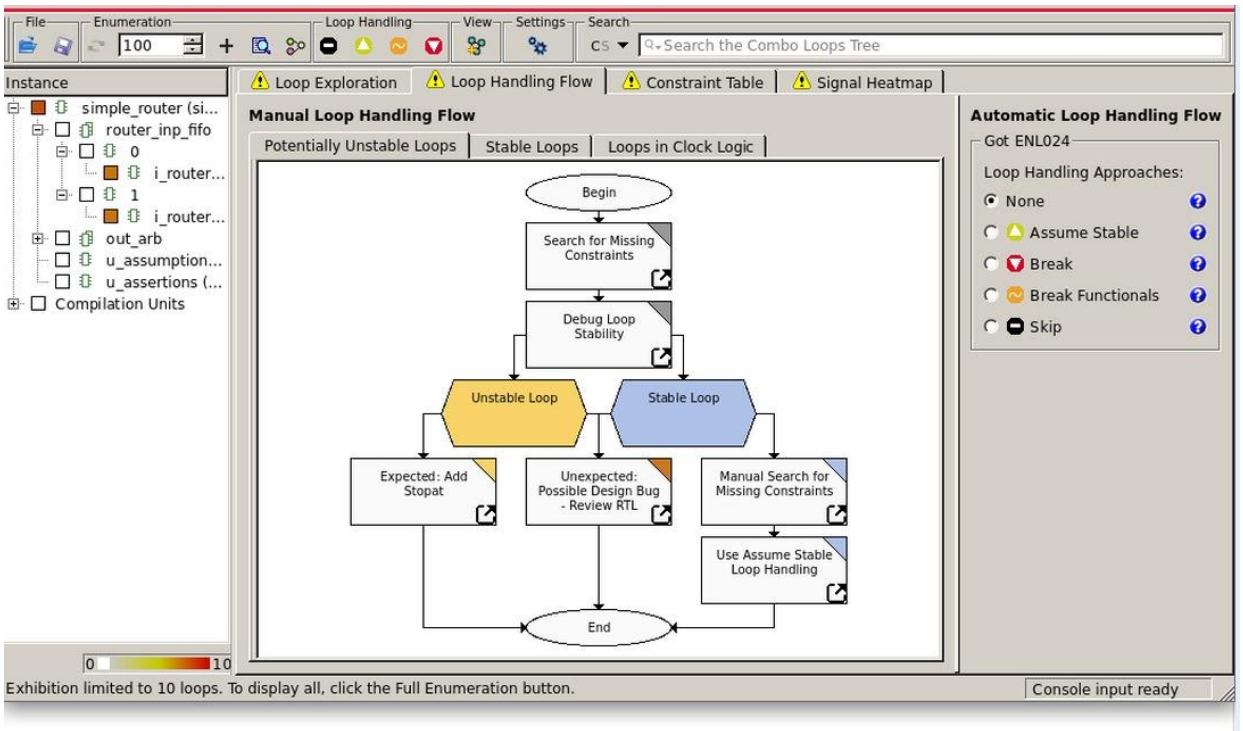


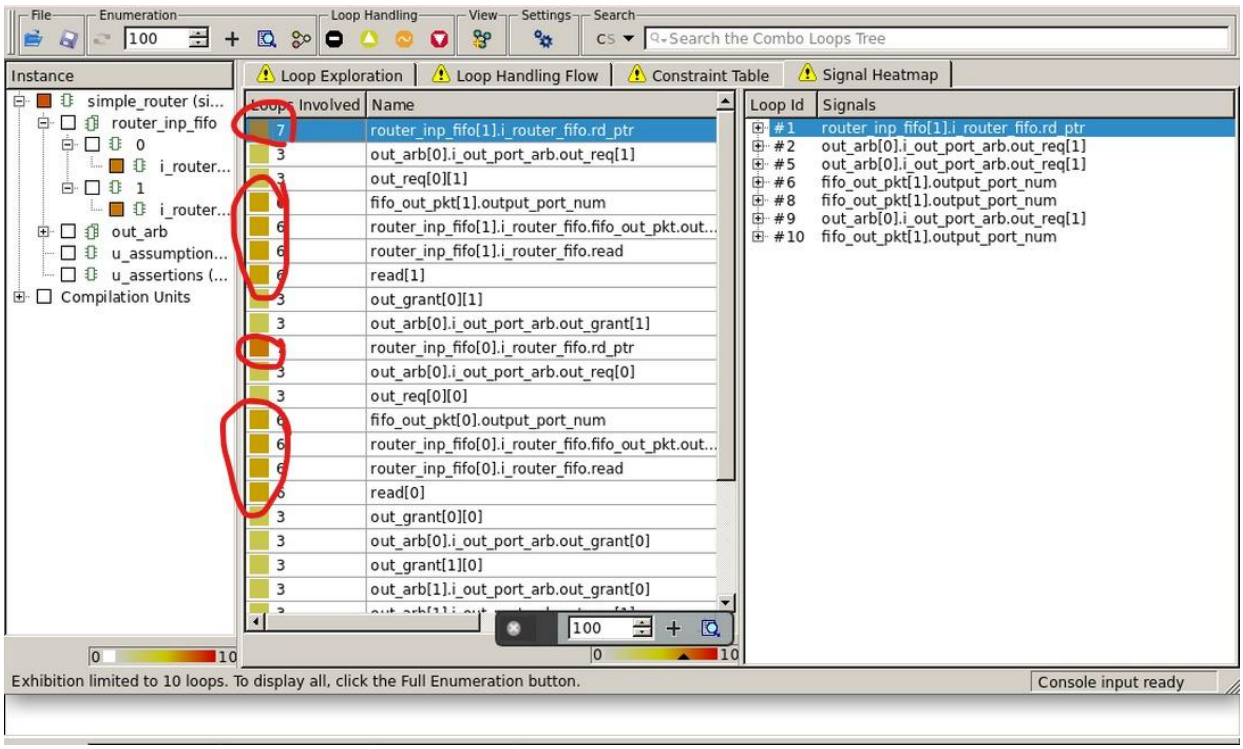
We are assigning a larger width bit to the smaller width target bit in the router fifo.

This warning means that some of my cover properties are written in a complex style, so Jasper is interpretting them using its own default behavior instead of the exact official SystemVerilog standard. Jasper suggests using the -lrm_cover_property option if strict standard behavior is required.

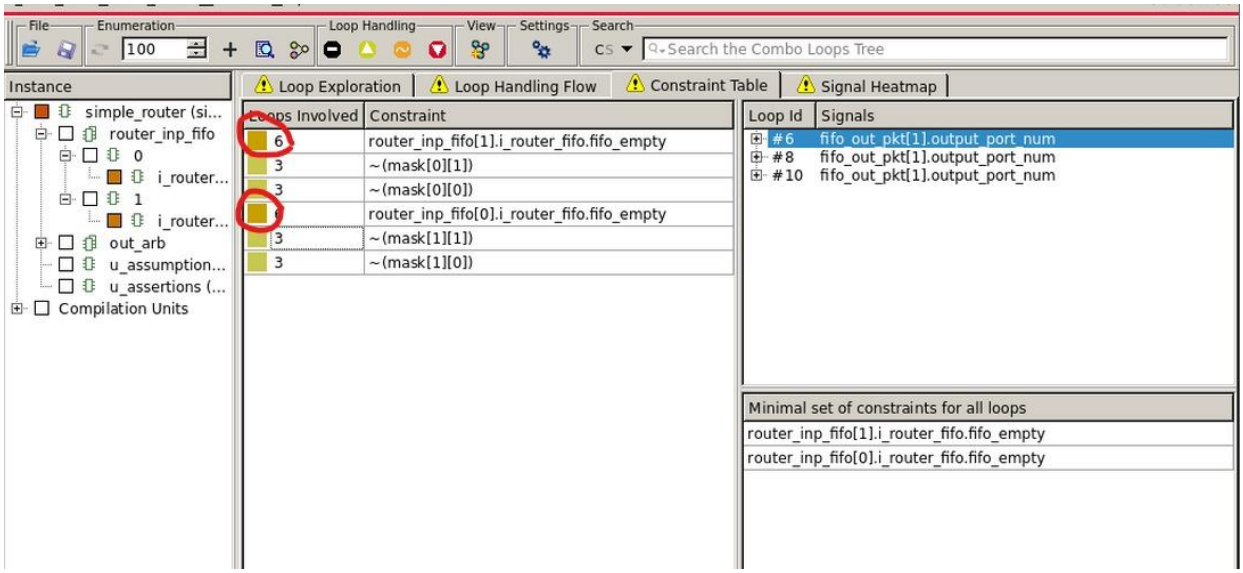
This is happening because the cover statements are not simple sequence-based covers, and the tool is just informing that coverage results may slightly differ from ideal System Verilog Standard defined semantics.


Applications Places System
Wed Nov 26, 18:23
(fifo_router_superlint) - Jasper Apps (.../run/jgproject) - Combo Loop Viewer
File Edit View Tools Window Help
cadence
Enumeration
Loop Handling
View
Settings
Search
Search the Combo Loops Tree
Instance
simple_router (si...
router_inp_fifo
0
i_router...
1
i_router...
out_arb
u_assumption...
u_assertions (...
Compilation Units
Loop Exploration
Loop Handling Flow
Constraint Table
Signal Heatmap
Manual Loop Handling Flow
Potentially Unstable Loops
Stable Loops
Loops in Clock Logic
Loop Id
Signals
#1 router_inp_fifo[1].i_router_fifo.rd_ptr
router_inp_fifo[1].i_router_fifo.rd_ptr
#2 out_arb[0].i_out_port_arb.out_req[1]
#3 router_inp_fifo[0].i_router_fifo.rd_ptr
#4 out_arb[0].i_out_port_arb.out_req[0]
#5 out_arb[0].i_out_port_arb.out_req[1]
#6 fifo_out_pkt[1].output_port_num
#7 fifo_out_pkt[0].output_port_num
#8 fifo_out_pkt[1].output_port_num
#9 out_arb[0].i_out_port_arb.out_req[1]
#10 fifo_out_pkt[1].output_port_num
Begin
Search for Mis
Constraint
Debug Loo
Stability
Unstable Loop
Expected: Add
Stopat
Unexpecte
Possible Desig
- Review RT
End
Automatic Loop Handling Flow
Got ENL024
Loop Handling Approaches:
None
Assume Stable
Break
Break Functionals
Skip
Exhibition limited to 10 loops. To display all, click the Full Enumeration button.
Console input ready
Console
Lint Messages
Warnings / Errors
Proof Messages
No proofs running
Console input ready
poojith
[Mate Terminal]
<Verdi-Apex:...
Mate Terminal
run_lint.tcl (fif...
(fifo_router_su...


File
Enumeration
Loop Handling
View
Settings
Search
Search the Combo Loops Tree
Instance
simple_router (si...
router_inp_fifo
0
i_router...
1
i_router...
out_arb
u_assumption...
u_assertions (...
Compilation Units
Loop Exploration
Loop Handling Flow
Constraint Table
Signal Heatmap
Manual Loop Handling Flow
Potentially Unstable Loops
Stable Loops
Loops in Clock Logic
Begin
Search for Missing Constraints
Debug Loop Stability
Unstable Loop
Stable Loop
Expected: Add Stopat
Unexpected: Possible Design Bug - Review RTL
Manual Search for Missing Constraints
Use Assume Stable Loop Handling
End
Automatic Loop Handling Flow
Got ENL024
Loop Handling Approaches:
None
Assume Stable
Break
Break Functionals
Skip
Exhibition limited to 10 loops. To display all, click the Full Enumeration button.
Console input ready

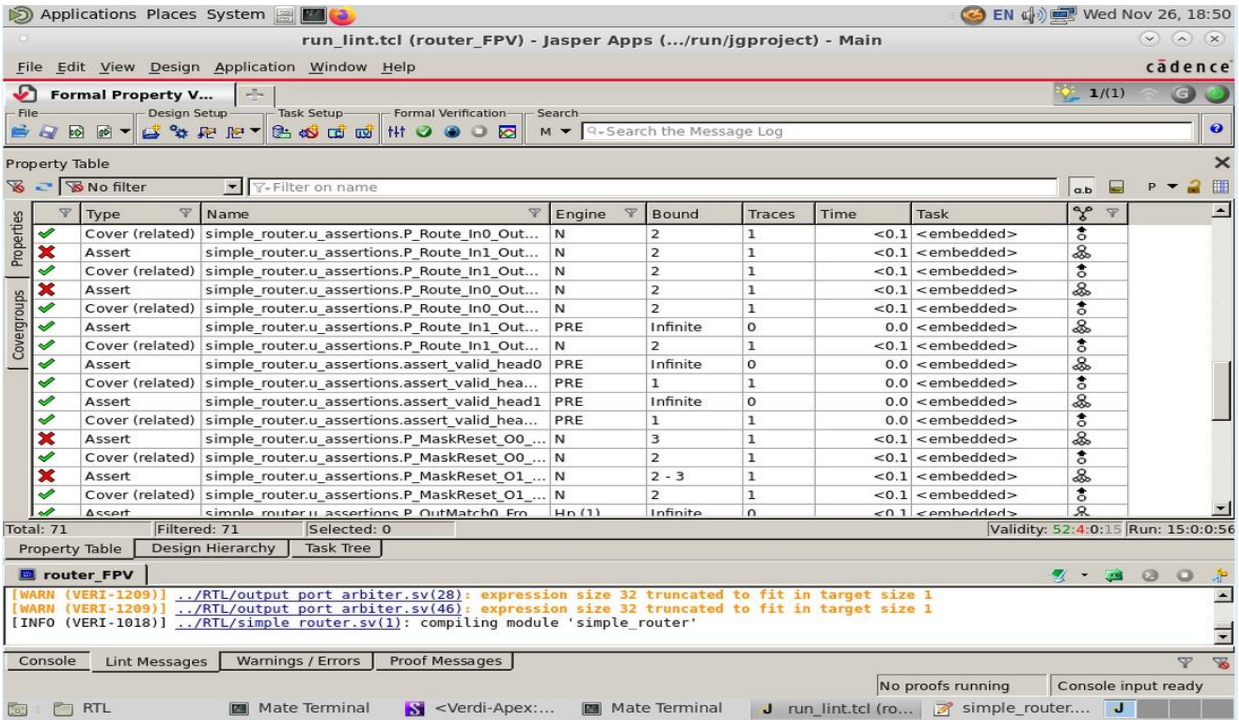


After carefully checking the signal heat map it indicates what are the signals involved in the combinational loop, this is the combinational loop we found in dynamic simulation where head and middle flits are repeating themselves as a valid output to either of outputs.

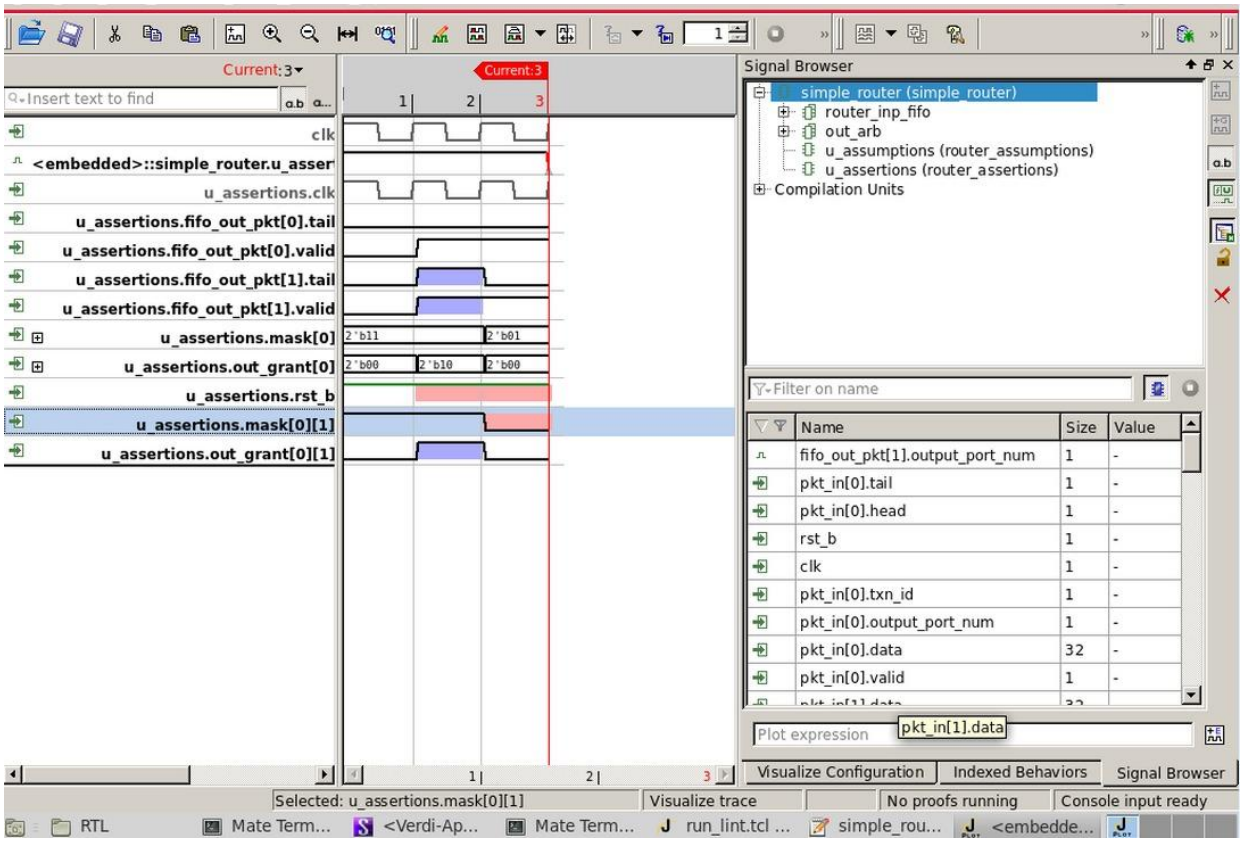


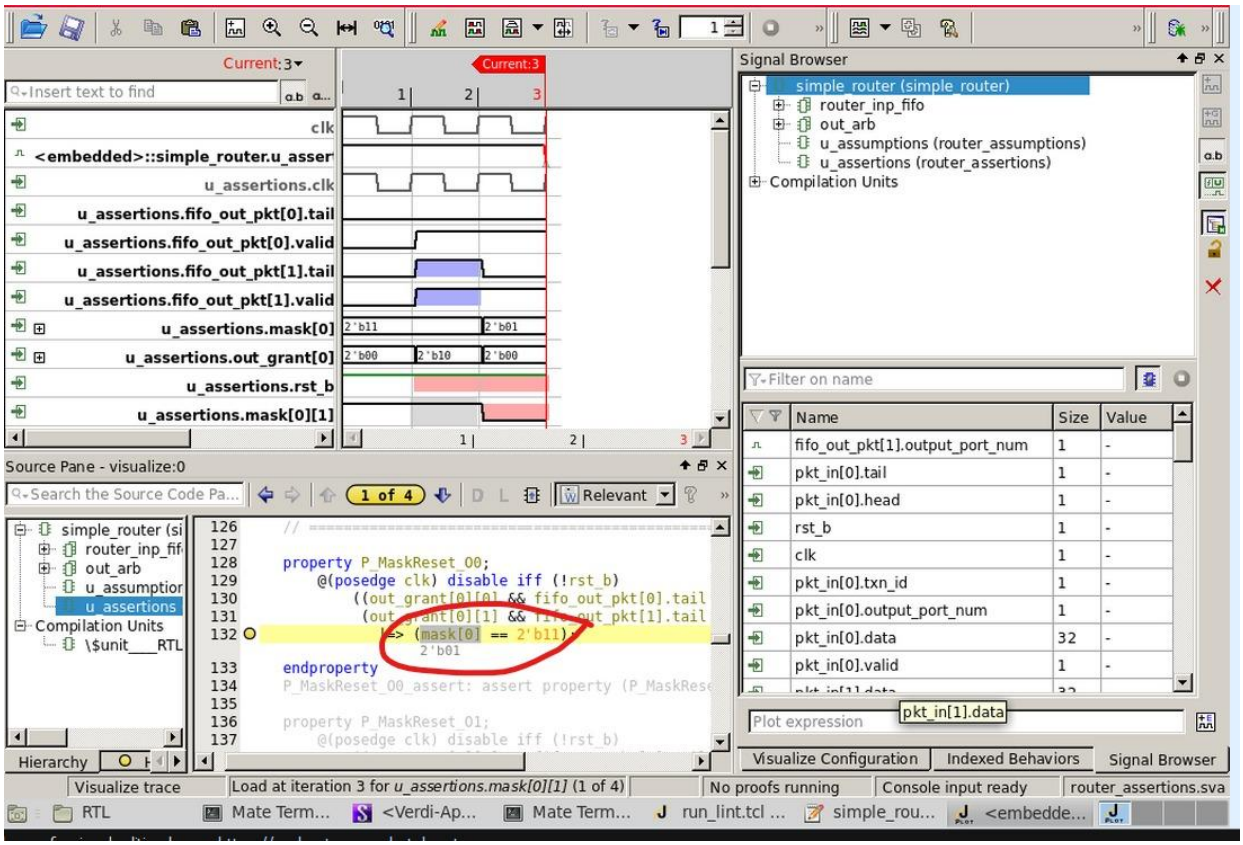
So, to tackle this we do not increase the read pointer if fifo is empty. It can be derived from constraint table as well.

Bug Identification1:



After resolving the above error the result, we got is four failed assertions.





So, mask bits of output[0] is resetting to 1 after receiving the tail bits from any of the inputs.

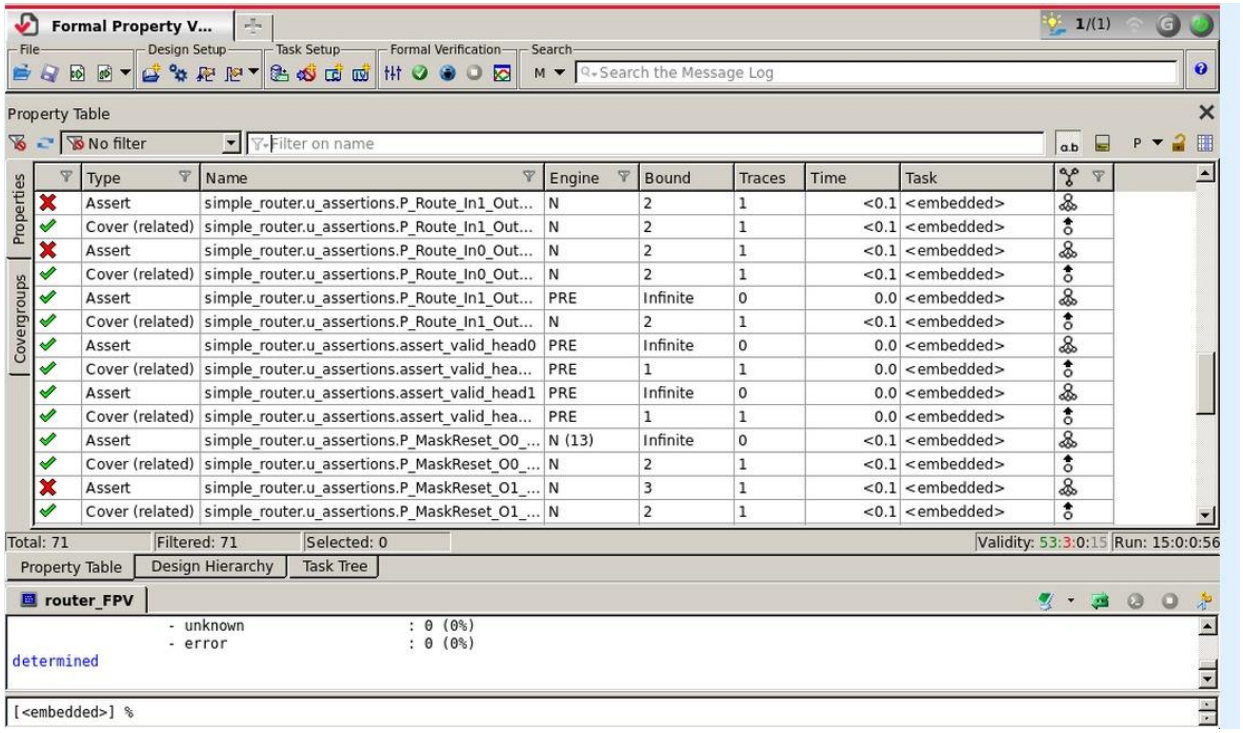
Before bug Fix:

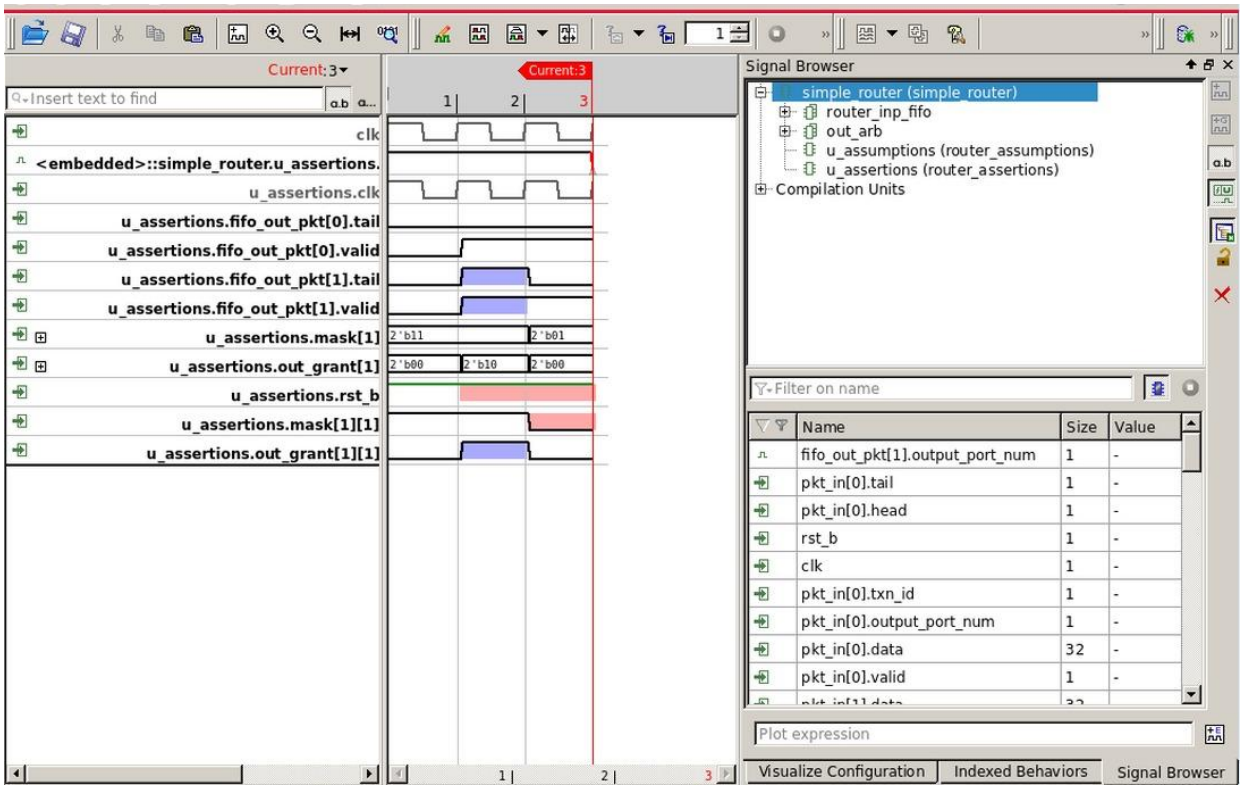
```
    end
    else begin
    if((out_grant[0][0] && fifo_out_pkt[0].tail) || (out_grant[0][1] && fifo_out_pkt[1].tail)) begin
        mask[0] <= 2'h1;     //before bug fix
    end
    else if (out_grant[0][0]) begin
        mask[0] <= 2'h1;
    end
    else if(out_grant[0][1]) begin
```

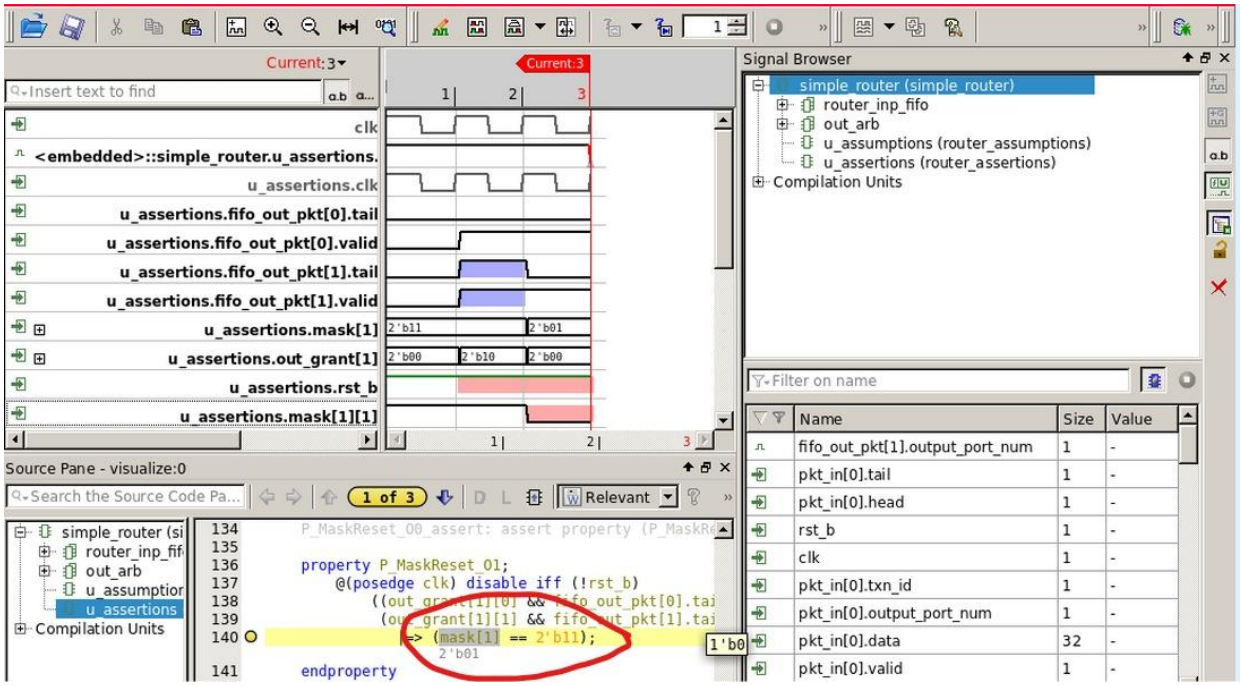
After bug Fix:

```
    mask[1] <= '1;
    end
    else begin
    if((out_grant[0][0] && fifo_out_pkt[0].tail) || (out_grant[0][1] && fifo_out_pkt[1].tail)) begin
        mask[0] <= 2'h3;     // bug fix
    end
    else if (out_grant[0][0]) begin
        mask[0] <= 2'h1;
    end
    else if(out_grant[0][1]) begin
```

Bug Identification2:







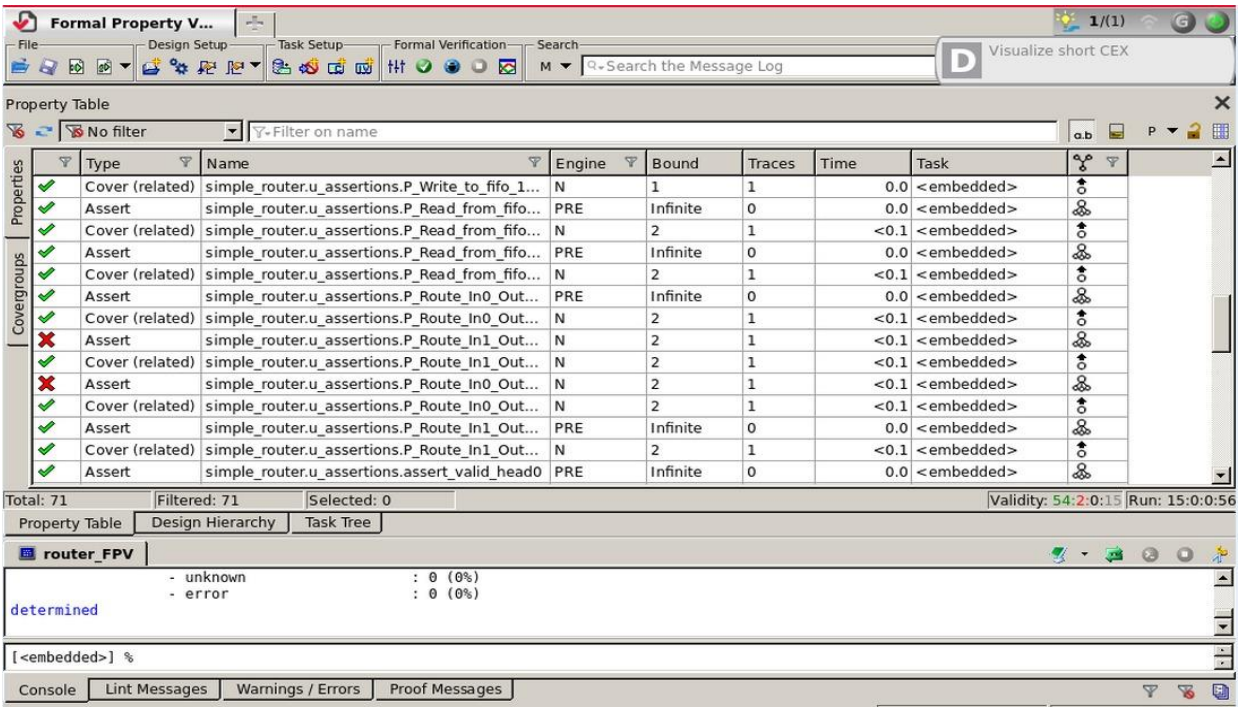
So, mask bits of output[1] is resetting to 1 after receiving the tail bits from any of the inputs.

Before Bug fix:

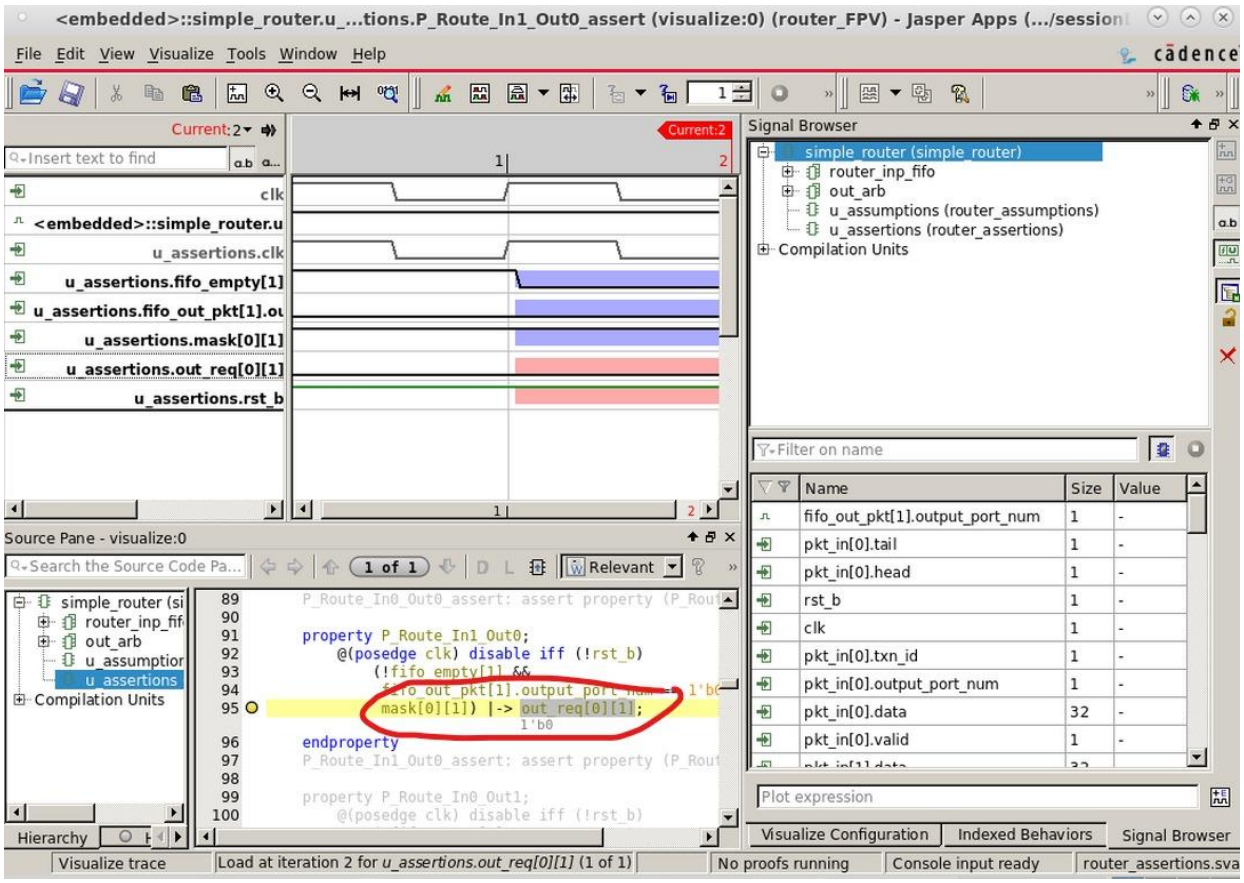
```
        mask[0] <= 2'h2;
    end

    if((out_grant[1][0] && fifo_out_pkt[0].tail) || (out_grant[1][1] && fifo_out_pkt[1].tail)) begin
        mask[1] <= 2'h1;    // bug
    end
    else if (out_grant[1][0]) begin
        mask[1] <= 2'h1;
    end
```

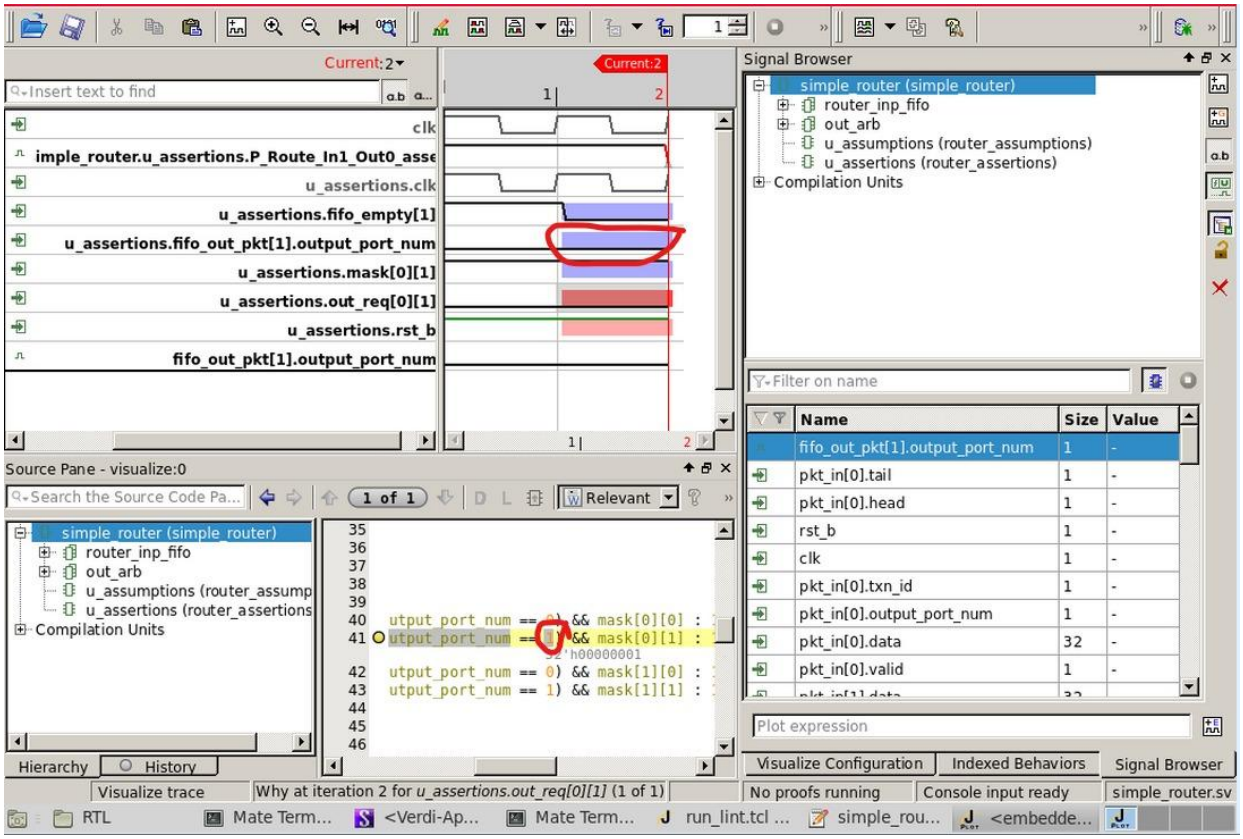
After Bug Fix:

```
    end

    if((out_grant[1][0] && fifo_out_pkt[0].tail) || (out_grant[1][1] && fifo_out_pkt[1].tail)) begin
        mask[1] <= 2'h1;    // bug fix
    end
    else if (out_grant[1][0]) begin
        mask[1] <= 2'h1;
    end
    else if(out_grant[1][1]) begin
```



Bug Identification3:





Output port is 0 but checking it against 1. So, the assertion failed for input[1] going to output[0]

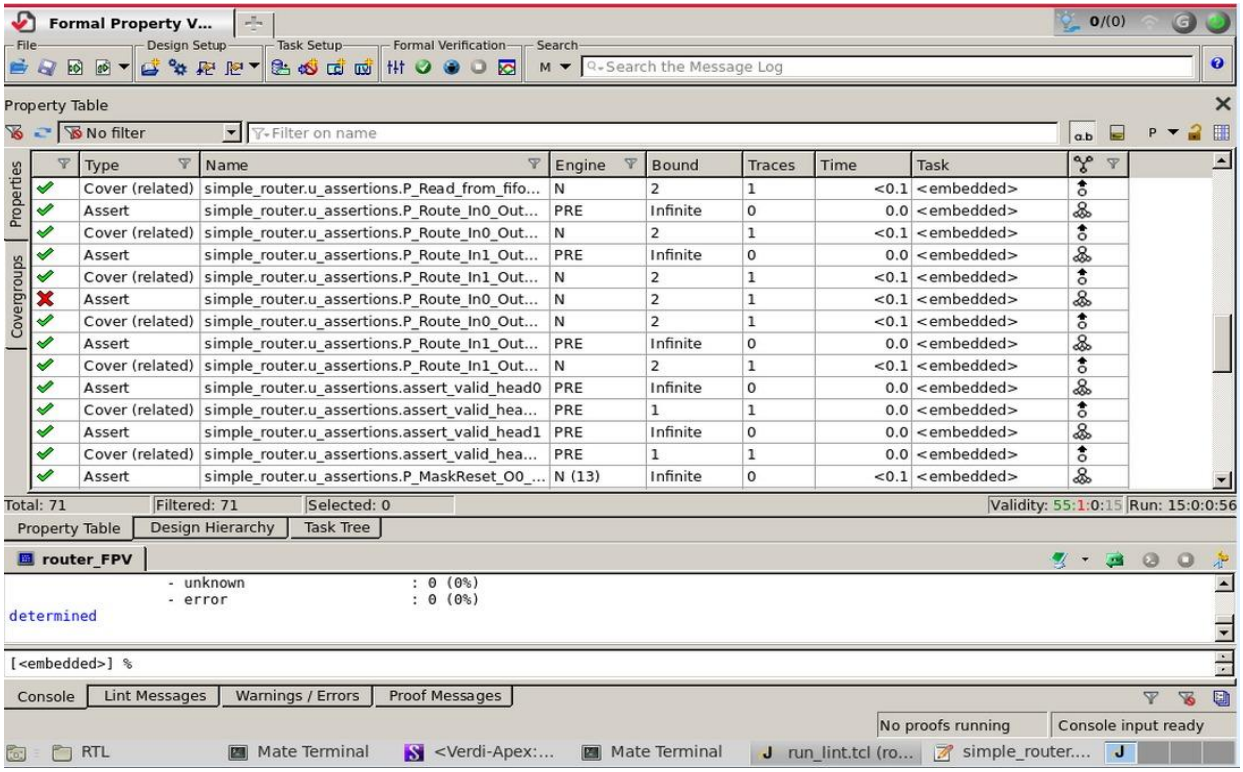
Before Bug Fix:

```
assign out_req[0][0] = !fifo_empty[0] ? (fifo_out_pkt[0].output_port_num == 0) && mask[0][0] : 1'b0;
assign out_req[0][1] = !fifo_empty[1] ? (fifo_out_pkt[1].output_port_num == 1) && mask[0][1] : 1'b0;    // bug
assign out_req[1][0] = !fifo_empty[0] ? (fifo_out_pkt[0].output_port_num == 0) && mask[1][0] : 1'b0;
assign out_req[1][1] = !fifo_empty[1] ? (fifo_out_pkt[1].output_port_num == 1) && mask[1][1] : 1'b0;
```

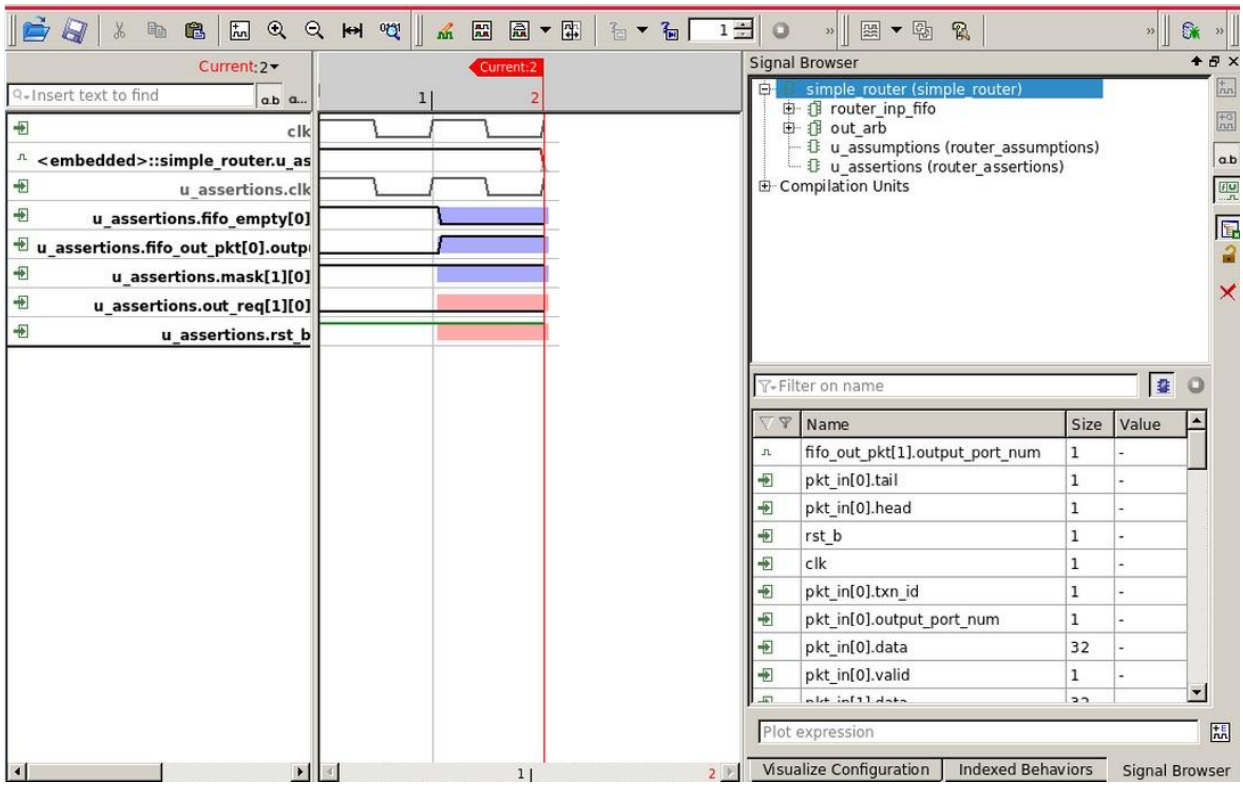
After Bug Fix:

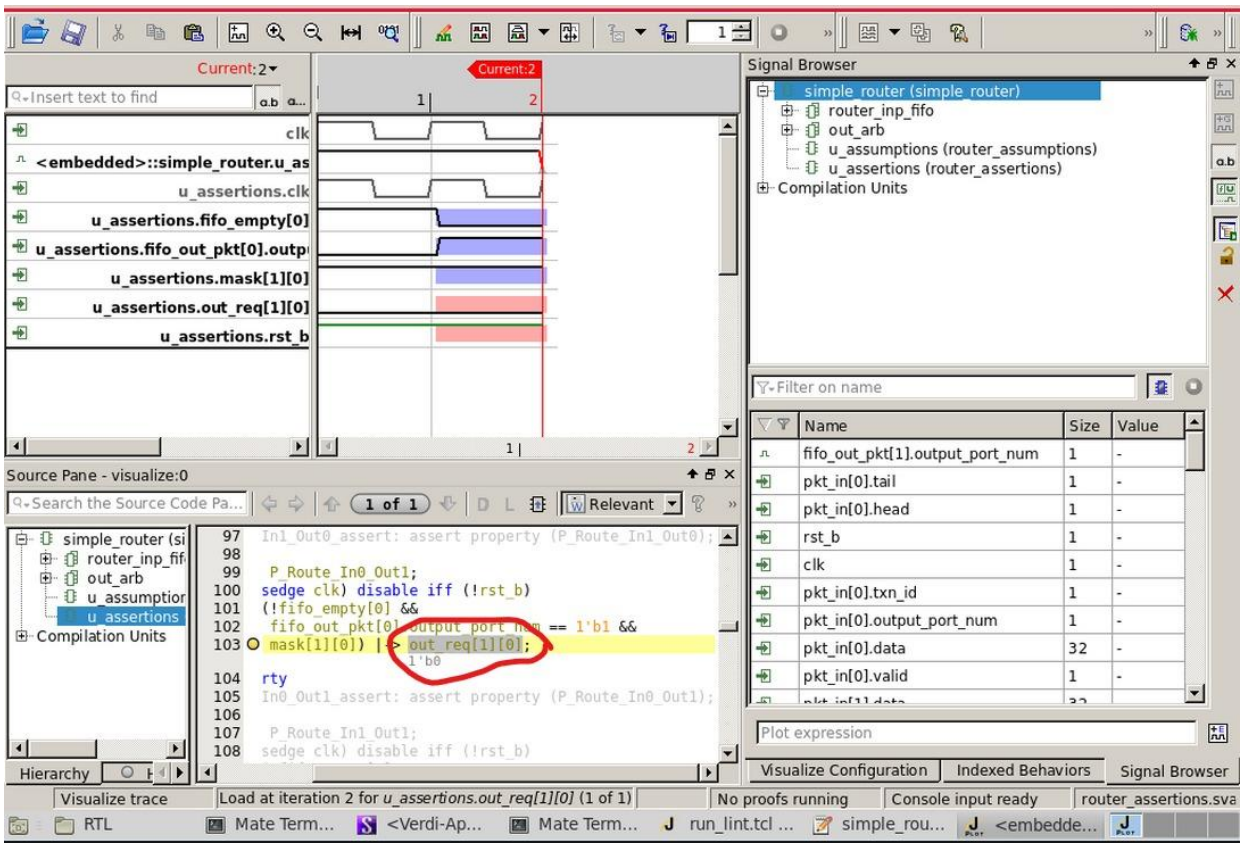
```
endgenerate

assign out_req[0][0] = !fifo_empty[0] ? (fifo_out_pkt[0].output_port_num == 0) && mask[0][0] : 1'b0;
assign out_req[0][1] = !fifo_empty[1] ? (fifo_out_pkt[1].output_port_num == 0) && mask[0][1] : 1'b0;    // bug fix
assign out_req[1][0] = !fifo_empty[0] ? (fifo_out_pkt[0].output_port_num == 0) && mask[1][0] : 1'b0;
assign out_req[1][1] = !fifo_empty[1] ? (fifo_out_pkt[1].output_port_num == 1) && mask[1][1] : 1'b0;
```

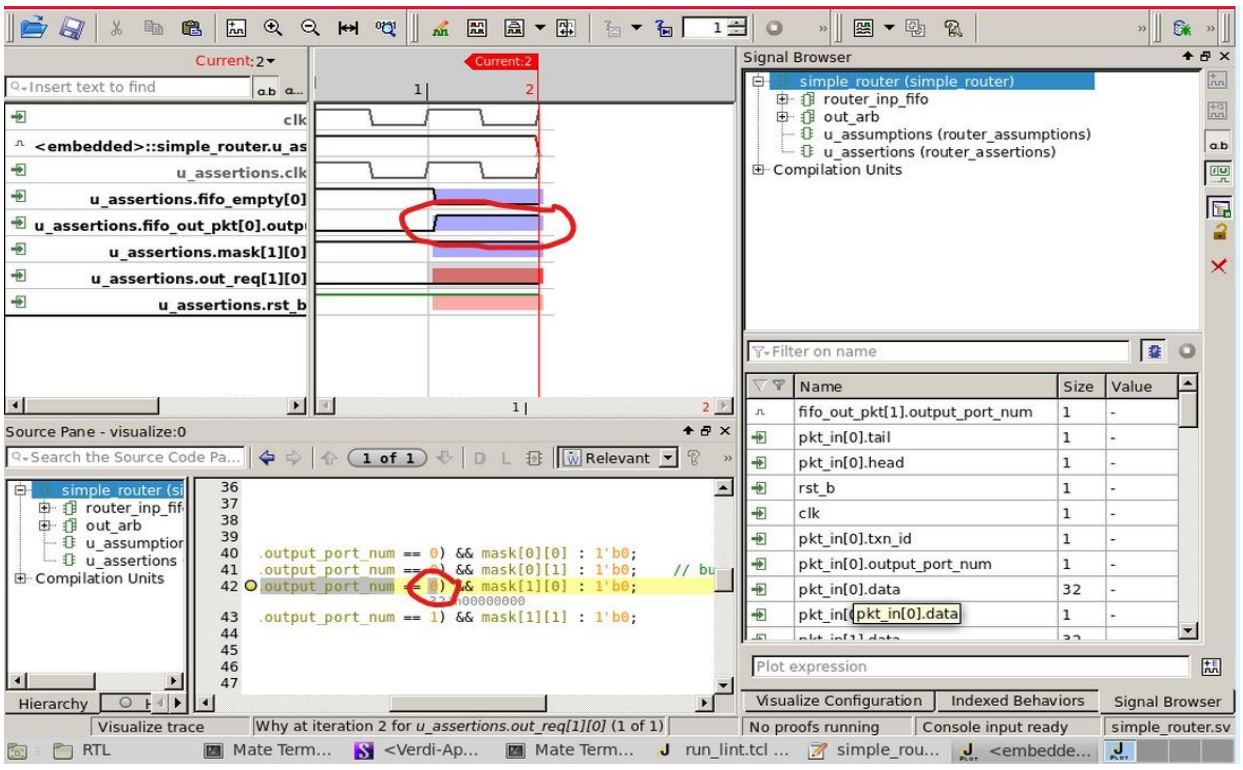
Bug Identification 4:



Type	Name	Engine	Bound	Traces	Time	Task
Cover (related)	simple_router.u_assertions.P_Read_from_fifo...	N	2	1	<0.1	<embedded>
Assert	simple_router.u_assertions.P_Route_In0_Out...	PRE	Infinite	0	0.0	<embedded>
Cover (related)	simple_router.u_assertions.P_Route_In0_Out...	N	2	1	<0.1	<embedded>
Assert	simple_router.u_assertions.P_Route_In1_Out...	PRE	Infinite	0	0.0	<embedded>
Cover (related)	simple_router.u_assertions.P_Route_In1_Out...	N	2	1	<0.1	<embedded>
Assert	simple_router.u_assertions.P_Route_In0_Out...	N	2	1	<0.1	<embedded>
Cover (related)	simple_router.u_assertions.P_Route_In0_Out...	N	2	1	<0.1	<embedded>
Assert	simple_router.u_assertions.P_Route_In1_Out...	PRE	Infinite	0	0.0	<embedded>
Cover (related)	simple_router.u_assertions.P_Route_In1_Out...	N	2	1	<0.1	<embedded>
Assert	simple_router.u_assertions.assert_valid_head0	PRE	Infinite	0	0.0	<embedded>
Cover (related)	simple_router.u_assertions.assert_valid_hea...	PRE	1	1	0.0	<embedded>
Assert	simple_router.u_assertions.assert_valid_head1	PRE	Infinite	0	0.0	<embedded>
Cover (related)	simple_router.u_assertions.assert_valid_hea...	PRE	1	1	0.0	<embedded>
Assert	simple_router.u_assertions.P_MaskReset_O0_...	N (13)	Infinite	0	<0.1	<embedded>







Output port is 1 but checking it against 0. So, the assertion failed for input[0] going to output[1]

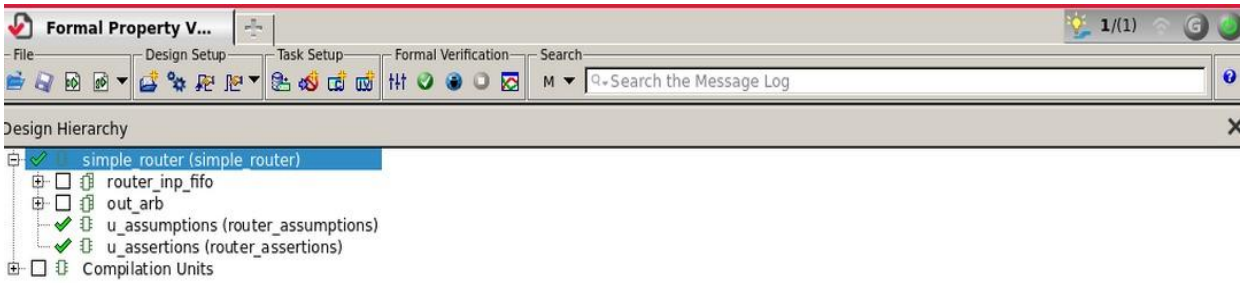
Before Bug Fix:

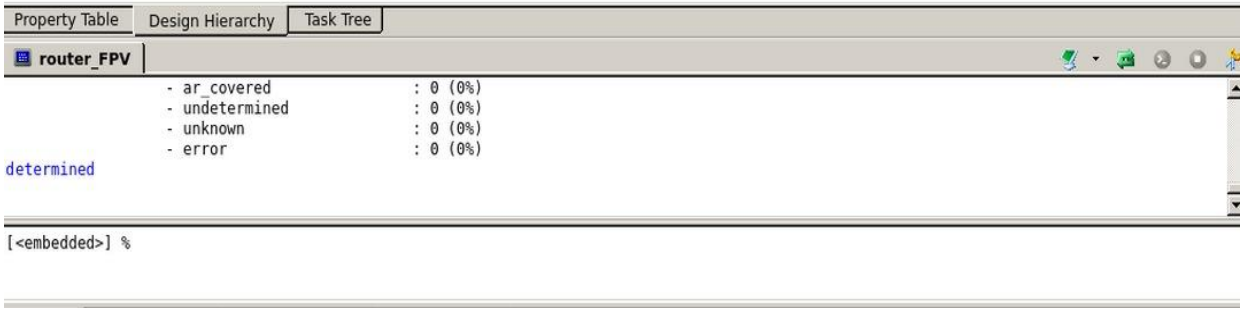
```
endgenerate

assign out_req[0][0] = !fifo_empty[0] ? (fifo_out_pkt[0].output_port_num == 0) && mask[0][0] : 1'b0;
assign out_req[0][1] = !fifo_empty[1] ? (fifo_out_pkt[1].output_port_num == 0) && mask[0][1] : 1'b0;    // bug fix
assign out_req[1][0] = !fifo_empty[0] ? (fifo_out_pkt[0].output_port_num == 0) && mask[1][0] : 1'b0;    // bug
assign out_req[1][1] = !fifo_empty[1] ? (fifo_out_pkt[1].output_port_num == 1) && mask[1][1] : 1'b0;

```

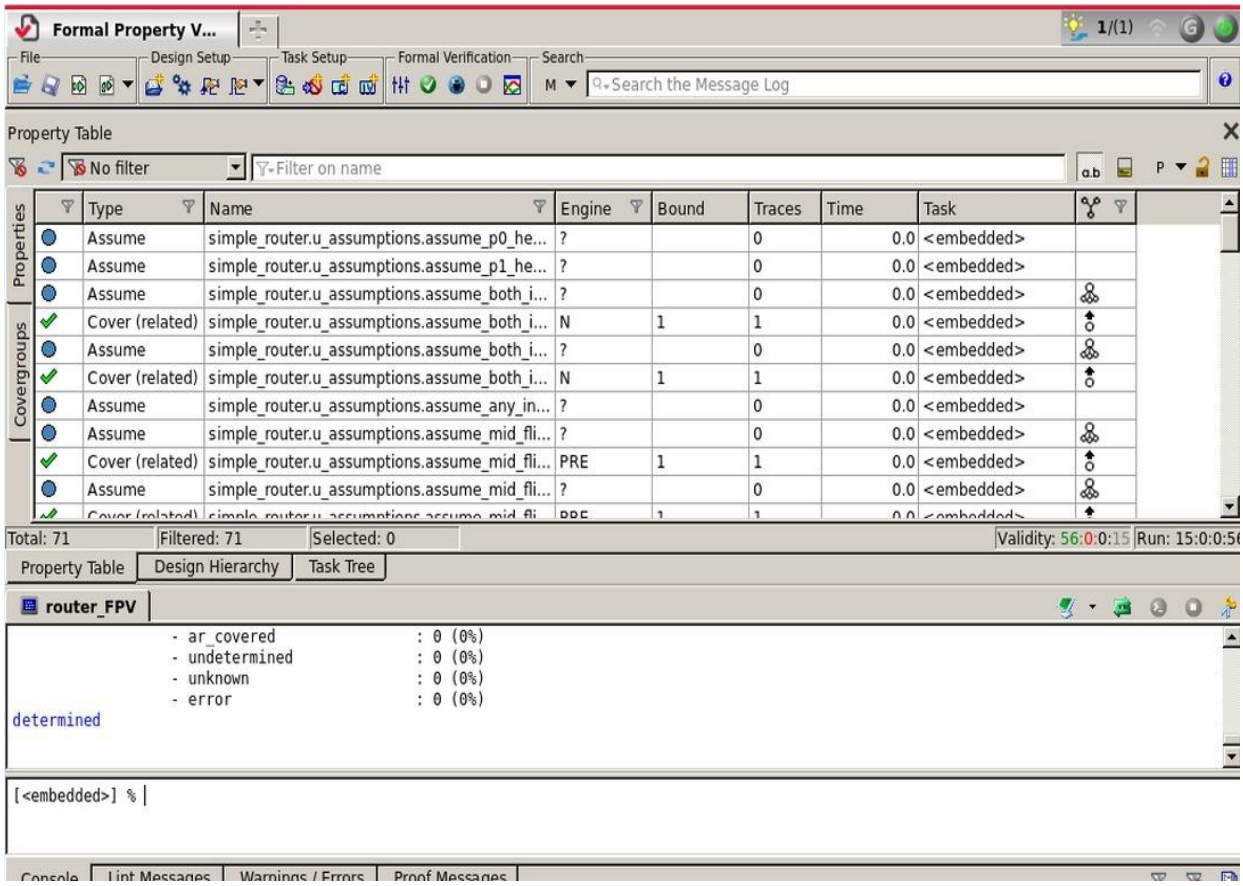
After Bug Fix:

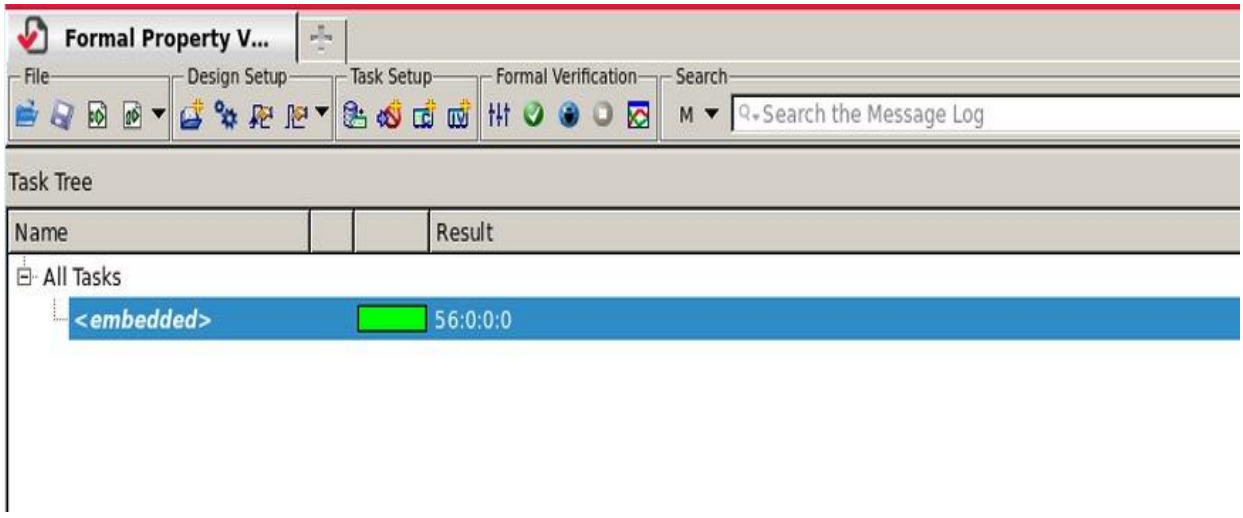
```
endgenerate

assign out_req[0][0] = !fifo_empty[0] ? (fifo_out_pkt[0].output_port_num == 0) && mask[0][0] : 1'b0;
assign out_req[0][1] = !fifo_empty[1] ? (fifo_out_pkt[1].output_port_num == 0) && mask[0][1] : 1'b0;    // bug fix
assign out_req[1][0] = !fifo_empty[0] ? (fifo_out_pkt[0].output_port_num == 0) && mask[1][0] : 1'b0;    // bug
assign out_req[1][1] = !fifo_empty[1] ? (fifo_out_pkt[1].output_port_num == 1) && mask[1][1] : 1'b0;

```

FPV Final Simulation Output:






Formal Property V...
1/(1)
File
Design Setup
Task Setup
Formal Verification
Search
Search the Message Log
Property Table
No filter
Filter on name
Properties
Covergroups
Type
Name
Engine
Bound
Traces
Time
Task
Assume simple_router.u_assumptions.assume_p0_he... ? 0 0.0 <embedded>
Assume simple_router.u_assumptions.assume_p1_he... ? 0 0.0 <embedded>
Assume simple_router.u_assumptions.assume_both_i... ? 0 0.0 <embedded>
Cover (related) simple_router.u_assumptions.assume_both_i... N 1 1 0.0 <embedded>
Assume simple_router.u_assumptions.assume_both_i... ? 0 0.0 <embedded>
Cover (related) simple_router.u_assumptions.assume_both_i... N 1 1 0.0 <embedded>
Assume simple_router.u_assumptions.assume_any_in... ? 0 0.0 <embedded>
Assume simple_router.u_assumptions.assume_mid_fli... ? 0 0.0 <embedded>
Cover (related) simple_router.u_assumptions.assume_mid_fli... PRE 1 1 0.0 <embedded>
Assume simple_router.u_assumptions.assume_mid_fli... ? 0 0.0 <embedded>
Total: 71
Filtered: 71
Selected: 0
Validity: 56:0:0:15
Property Table
Design Hierarchy
Task Tree
router_FPV
- ar_covered : 0 (0%)
- undetermined : 0 (0%)
- unknown : 0 (0%)
- error : 0 (0%)
determined
[<embedded>] %
Console
Lint Messages
Warnings / Errors
Proof Messages


Formal Property V...
File
Design Setup
Task Setup
Formal Verification
Search
Search the Message Log
Task Tree
Name
Result
All Tasks
<embedded>
56:0:0:0

Show usage of different app other than FPV. Explain what app did you use? Why you chose that and how did it help your verification.

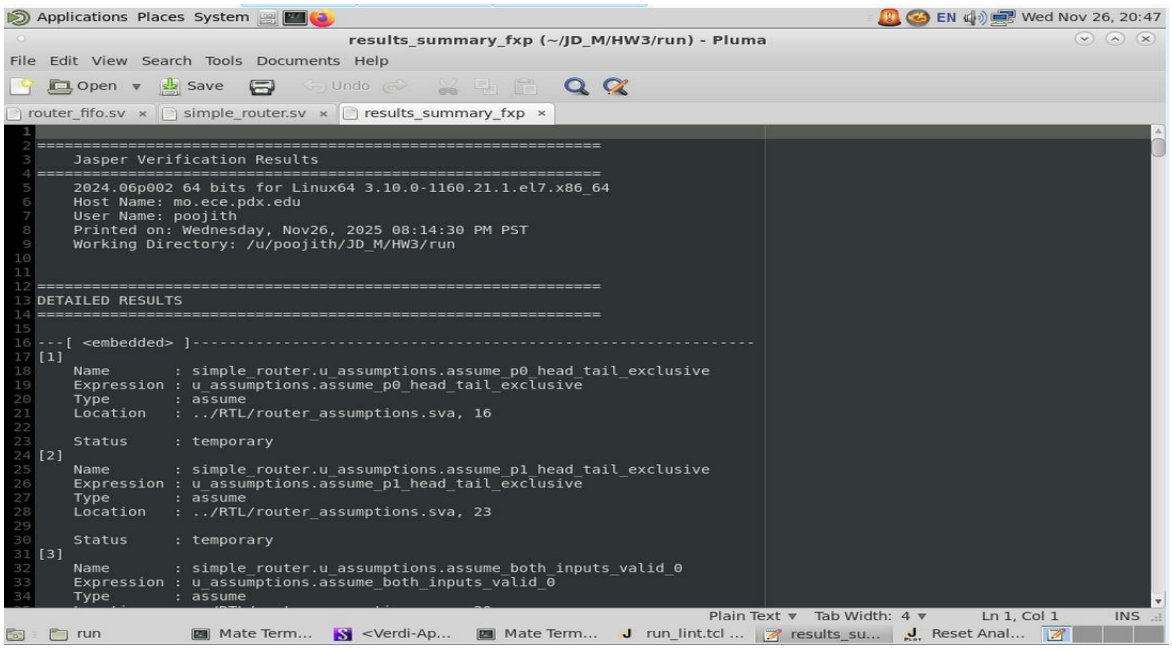
After completing functional verification using the FPV application in JasperGold, Again I have chosen to use the FXP (Formal X-Propagation) application to analyze how unknown values (X-states) propagate through the router logic. The purpose was to evaluate the design's robustness during reset, masking, FIFO operation, and arbitration control.

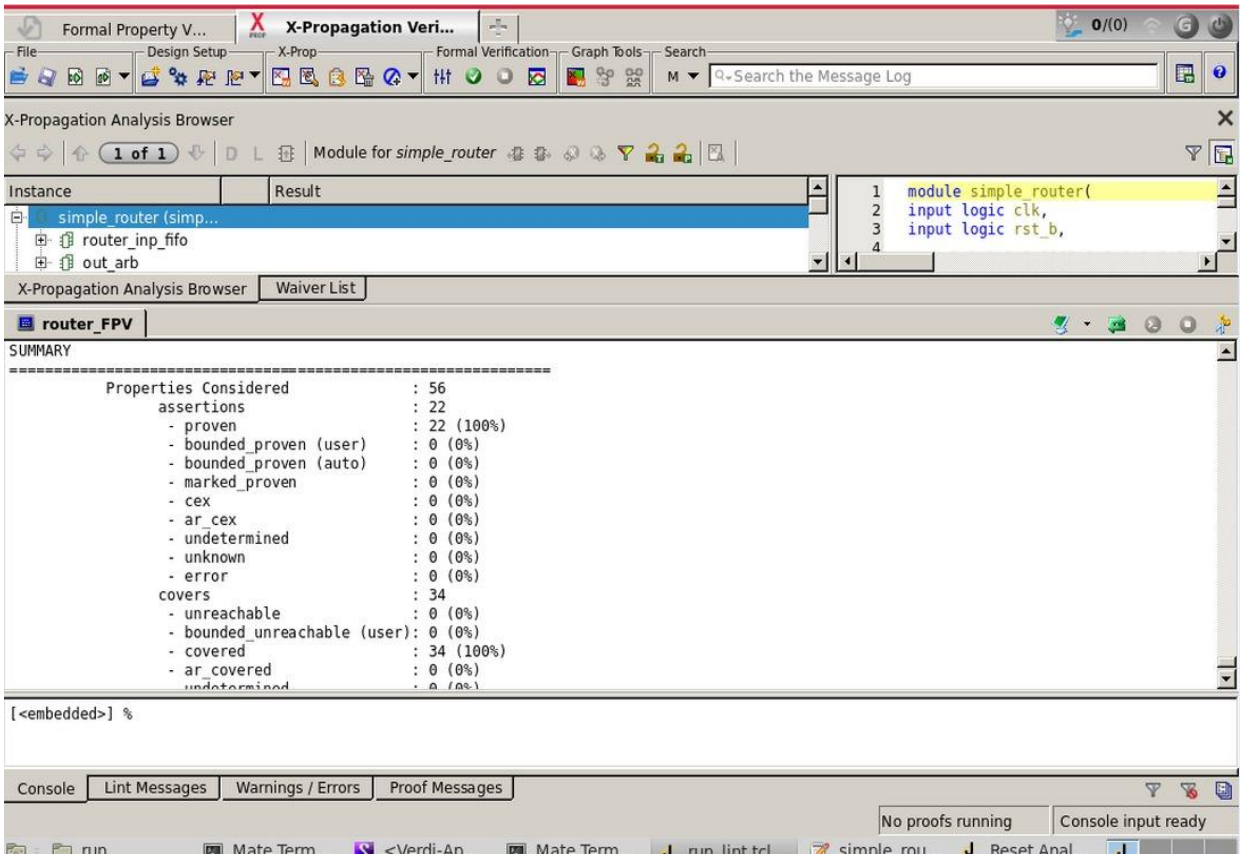
Same as mentioned in VC formal FXP usage it plays a crucial role in identifying hidden design flaws that were not easily visible in standard simulation. It exposed problems related to reset initialization, request generation, masking logic, FIFO management, and arbitration behavior.

Why we have opted for only FXP is that in other apps like SEQ and AEP, SEQ is mainly used for sequential equivalence checking between two versions of a design, such as a specification model and its implementation. Since only a single RTL version of the router was available and no golden reference existed, SEQ was not applicable.

Similarly, AEP (Assertion Execution Platform) is primarily focused on proving assertion correctness and coverage, but it does not specifically target the analysis of unknown (X) value propagation through the design. Our main concern was to identify how X-states affect mask logic, FIFO behavior, arbitration, and output data flow, which AEP does not explicitly address. Therefore, FXP was the most suitable choice as it is specifically designed to detect, trace, and analyze X-propagation issues, helping improve the robustness and reliability of the router design beyond just functional correctness.

FXP Simulation Output(Jasper Gold):




Formal Property V...
X-Propagation Veri...
0/(0)
File
Design Setup
X-Prop
Formal Verification
Graph Tools
Search
Search the Message Log
X-Propagation Analysis Browser
1 of 1
Module for simple_router
Instance
Result
simple_router (simp...
router_inp_fifo
out_arb
1 module simple_router(
2 input logic clk,
3 input logic rst_b,
X-Propagation Analysis Browser
Waiver List
router_FPV
SUMMARY
Properties Considered : 56
assertions : 22
- proven : 22 (100%)
- bounded_proven (user) : 0 (0%)
- bounded_proven (auto) : 0 (0%)
- marked_proven : 0 (0%)
- cex : 0 (0%)
- ar_cex : 0 (0%)
- undetermined : 0 (0%)
- unknown : 0 (0%)
- error : 0 (0%)
covers : 34
- unreachable : 0 (0%)
- bounded_unreachable (user): 0 (0%)
- covered : 34 (100%)
- ar_covered : 0 (0%)
[<embedded>] %
Console
Lint Messages
Warnings / Errors
Proof Messages
No proofs running
Console input ready