

## Dynamic and Formal Verification of 2x2 ROUTER

Poojith Pogarthi  
Sri Chandana Reddy Vanukuri

---

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

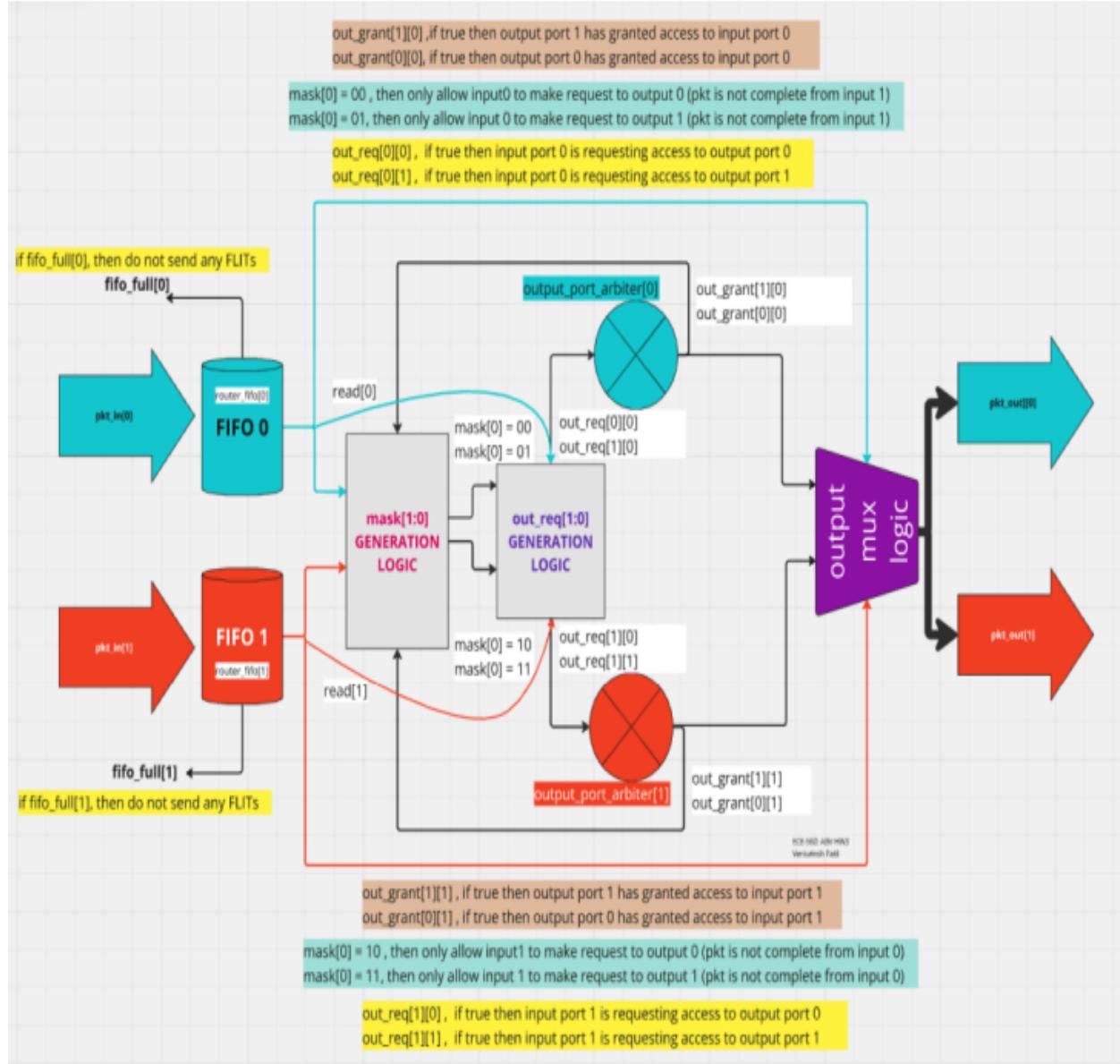
**We are considering the specification 1 rules and starting the verification process for the simple 2x2 router design.**

### Specification 1:

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

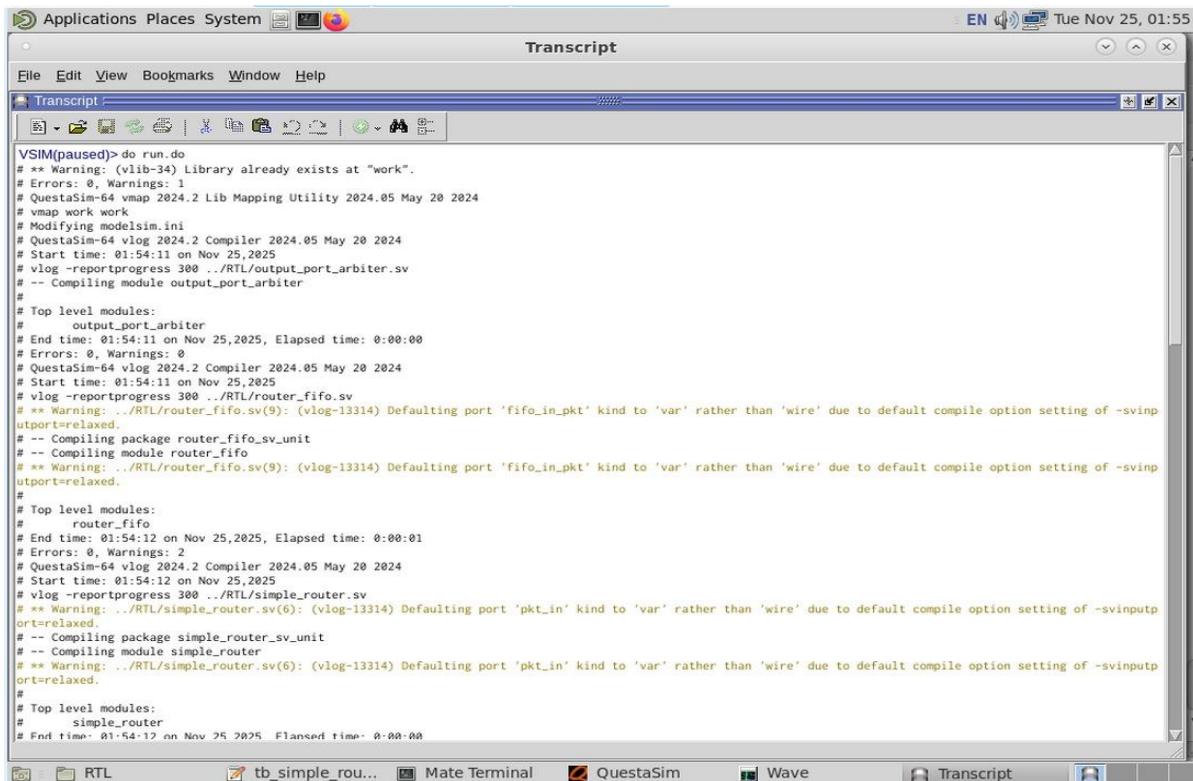
- if an output port arbiter gives grant to an input, then until we have sent out the entire packet (head flit, middle flit and tail flit) out from the same input port, we should not give grant to another input port
- Example: if `out_grant[0][1]` is 1 , that means output port 0 gave grant to input port 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.



The screenshot shows the QuestaSim Transcript window with the following log output:

```
VSIM(paused)> do run.do
# ** Warning: (vlib-34) Library already exists at "work".
# Errors: 0, Warnings: 1
QuestaSim-64 vmap 2024.2 Lib Mapping Utility 2024.05 May 20 2024
vmap work work
Modifying modelsim.ini
QuestaSim-64 vlog 2024.2 Compiler 2024.05 May 20 2024
Start time: 01:54:11 on Nov 25, 2025
vlog -reportprogress 300 ..RTL/output_port_arbiter.sv
-- Compiling module output_port_arbiter
#
# Top level modules:
#   output_port_arbiter
# End time: 01:54:11 on Nov 25, 2025, Elapsed time: 0:00:00
# Errors: 0, Warnings: 0
QuestaSim-64 vlog 2024.2 Compiler 2024.05 May 20 2024
Start time: 01:54:11 on Nov 25, 2025
vlog -reportprogress 300 ..RTL/router_fifo.sv
# ** Warning: ..RTL/router_fifo.sv(9): (vlog-13314) Defaulting port 'fifo_in_pkt' kind to 'var' rather than 'wire' due to default compile option setting of -svinputp
#ort=relaxed.
# -- Compiling package router_fifo_sv_unit
# -- Compiling module router_fifo
# ** Warning: ..RTL/router_fifo.sv(9): (vlog-13314) Defaulting port 'fifo_in_pkt' kind to 'var' rather than 'wire' due to default compile option setting of -svinputp
#ort=relaxed.
#
# Top level modules:
#   router_fifo
# End time: 01:54:12 on Nov 25, 2025, Elapsed time: 0:00:01
# 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/simple_router.sv
# ** Warning: ..RTL/simple_router.sv(b): (vlog-13314) Defaulting port 'pkt_in' kind to 'var' rather than 'wire' due to default compile option setting of -svinputp
#ort=relaxed.
# -- Compiling package simple_router_sv_unit
# -- Compiling module simple_router
# ** Warning: ..RTL/simple_router.sv(6): (vlog-13314) Defaulting port 'pkt_in' kind to 'var' rather than 'wire' due to default compile option setting of -svinputp
#ort=relaxed.
#
# Top level modules:
#   simple_router
# End time: 01:54:12 on Nov 25, 2025, Elapsed time: 0:00:00
```

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(fast)
# Loading work.router_fifo_sv_unit(fast)
# Loading work.router_fifo(fast)
# Loading work.output_port_arbiter(fast)
# T=5000 | 1st | IN0: v h t data      op txm | IN1: v h t data      op txm | OUT0: v h t data      | OUT1: v h t data
# T=5000 |   0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0
# T=15000 |   0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0
# T=25000 |   0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0
# T=35000 |   1 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0
# ==input port 0 to output port 0=====
# T=45000 |   1 | 1 0 0 0000001 0 0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0
# T=55000 |   1 | 1 0 0 0000002 0 0 | 0 0 0 0000000 0 0 | 1 1 0 0 0000001 0 0 | 1 1 0 0 0000001 0 0
# T=65000 |   1 | 1 0 0 0000003 0 0 | 0 0 0 0000000 0 0 | 1 0 0 0 0000002 0 0 | 1 0 0 0 0000002 0 0
# T=75000 |   1 | 1 0 1 0 0000004 0 0 | 0 0 0 0000000 0 0 | 1 0 0 0 0000003 0 0 | 1 0 0 0 0000003 0 0
# T=85000 |   1 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 | 1 0 1 0 0000004 0 0 | 1 0 1 0 0000004 0 0
# ==input port 1 to output port 0=====
# T=95000 |   1 | 0 0 0 0000000 0 0 | 1 1 0 0 0000005 0 1 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0
# T=105000 |   1 | 0 0 0 0000000 0 0 | 1 0 0 0 0000006 0 1 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0
# T=115000 |   1 | 0 0 0 0000000 0 0 | 1 0 0 0 0000007 0 1 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0
```

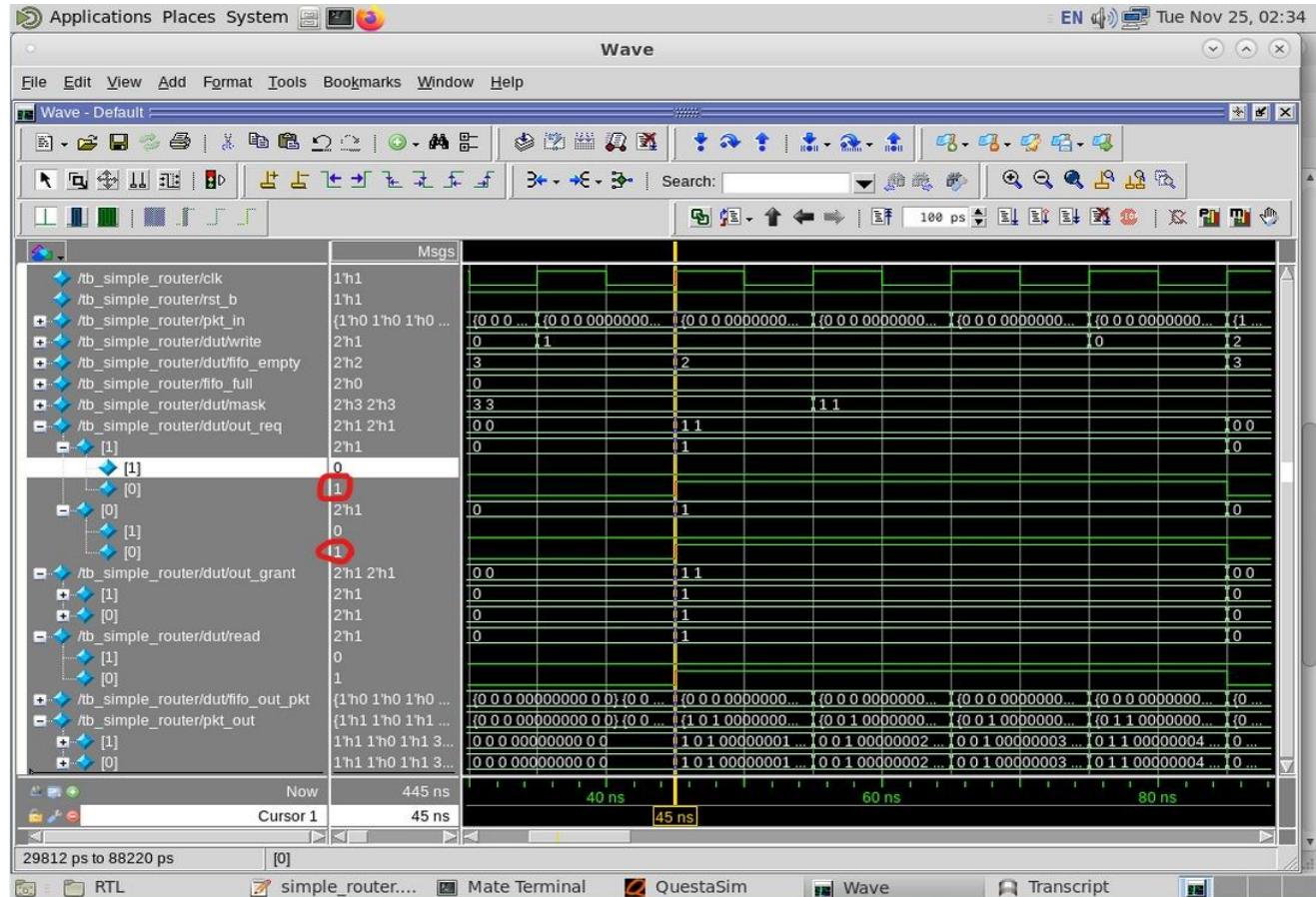
Transcript

```
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, input port 0 to output port 0=====
# 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
# =====
```

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.

```

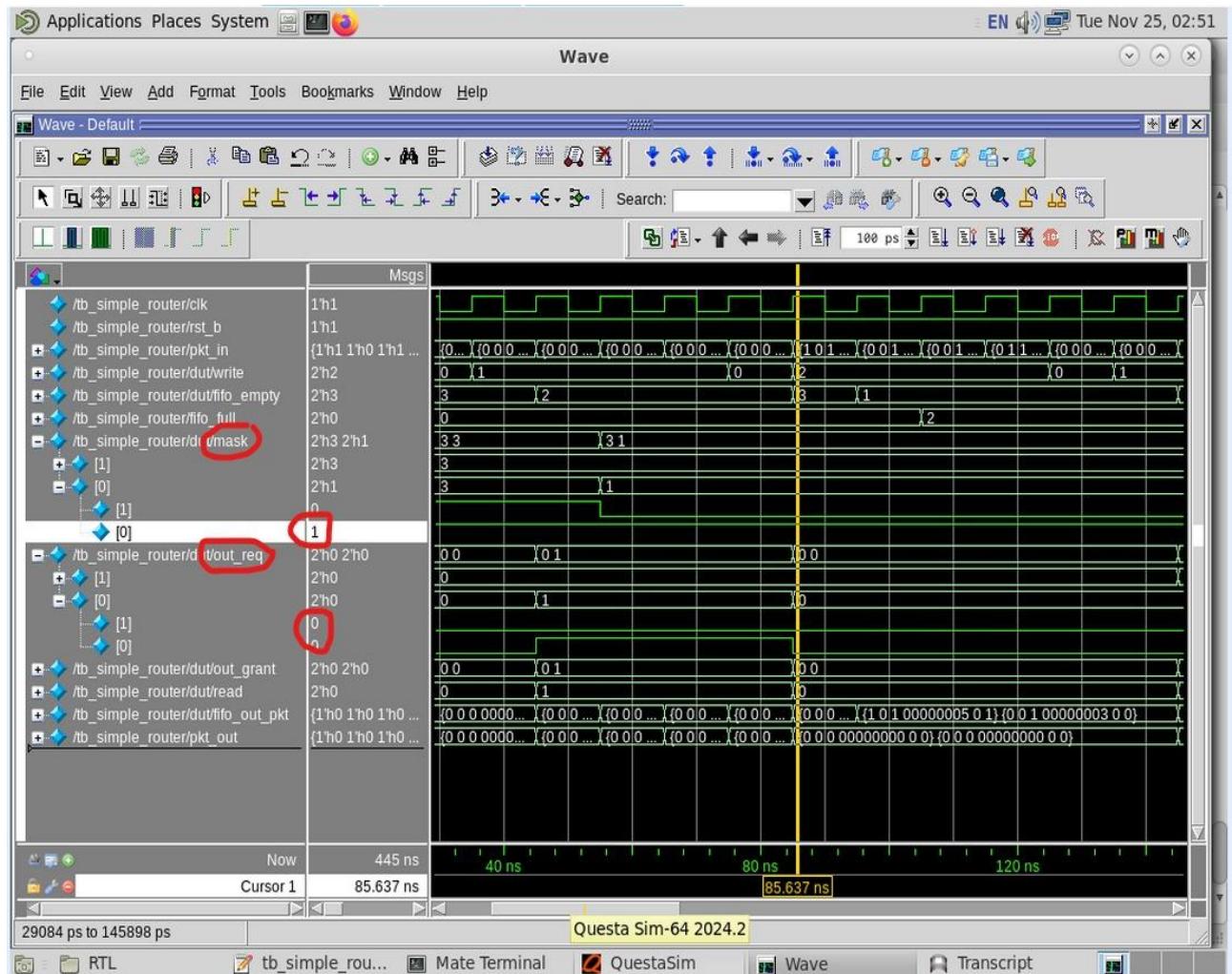
Transcript
File Edit View Bookmarks Window Help
Transcript
File Edit View Bookmarks Window Help
# ** Note: (vsim-3813) Design is being optimized due to module recompilation...
# ** Warning: (vopt-10587) Some optimizations are turned off because the +acc switch is in effect. This will cause your simulation to run slowly. Please use -accs s/-debug to maintain needed visibility. The +acc switch would be deprecated in a future release.
# ** Warning: ./RTL/simple_router.sv(6): (vopt-13314) Defaulting port 'pkt_in' kind to 'var' rather than 'wire' due to default compile option setting of -svinputp ort=relaxed.
# ** Note: (vsim-12126) Error and warning message counts have been restored: Errors=0, Warnings=2.
# Loading sv_std.svh
# 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 0000000 0 0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 |
# T=15000 | 0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 |
# T=25000 | 0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 |
# T=35000 | 1 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 |
# ==input port 0 to output port 0=====
# T=45000 | 1 | 1 0 0 0000001 0 0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 |
# T=55000 | 1 | 1 0 0 0000002 0 0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 | 1 1 0 0000001 0 0 |
# T=65000 | 1 | 1 0 0 0000003 0 0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 | 1 0 0 0000002 0 0 |
# T=75000 | 1 | 1 0 1 0000004 0 0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 | 1 0 0 0000003 0 0 |
# T=85000 | 1 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 | 1 0 1 0000004 0 0 |
# ==input port 1 to output port 0=====
# T=95000 | 1 | 0 0 0 0000000 0 0 | 1 1 0 0000005 0 1 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 |
# T=105000 | 1 | 0 0 0 0000000 0 0 | 1 0 0 0000006 0 1 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 |
# T=115000 | 1 | 0 0 0 0000000 0 0 | 1 0 0 0000007 0 1 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 |
# T=125000 | 1 | 0 0 0 0000000 0 0 | 1 0 1 0000008 0 1 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 |
# T=135000 | 1 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 1 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 |
# ==input port 0 to output port 1=====
# T=145000 | 1 | 1 1 0 0000001 1 0 | 0 0 0 0000000 0 1 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 |
# T=155000 | 1 | 1 0 0 0000002 1 0 | 0 0 0 0000000 0 1 | 0 0 0 0000000 0 0 | 1 1 0 0000001 0 0 |
# T=165000 | 1 | 1 0 1 0000004 1 0 | 0 0 0 0000000 0 1 | 0 0 0 0000000 0 0 | 1 0 0 0000002 0 0 |
# T=175000 | 1 | 0 0 0 0000000 1 0 | 0 0 0 0000000 0 1 | 0 0 0 0000000 0 0 | 1 0 1 0000004 0 0 |
# ==input port 1 to output port 1=====
# T=185000 | 1 | 0 0 0 0000000 1 0 | 1 1 0 0000001 1 1 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 |
# T=195000 | 1 | 0 0 0 0000000 1 0 | 1 0 0 0000002 1 1 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 |
# T=205000 | 1 | 0 0 0 0000000 1 0 | 1 0 1 0000004 1 1 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 |
# T=215000 | 1 | 0 0 0 0000000 1 0 | 0 0 0 0000000 1 1 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 |
# ==input port 1 to output port 1, input port 0 to output port 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  $\text{mask}[0] = 01$ . 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  $\text{mask}[0] = 01$  to  $\text{mask}[0] = 11$ , 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;
64     end

```

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

```

# 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 tx | IN1: v h t data    op tx | OUT0: v h t data    | OUT1: v h t data
# T=5000 | 0   | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 | 0 0 0 0000000
# T=15000 | 0   | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 | 0 0 0 0000000
# T=25000 | 0   | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 | 0 0 0 0000000
# T=35000 | 1   | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 | 0 0 0 0000000
# ==input port 0 to output port 0=====
# T=45000 | 1   | 1 0 0 0000001 0 0 | 0 0 0 0000000 0 0 | 0 0 0 0000000 | 0 0 0 0000000
# T=55000 | 1   | 1 0 0 0000002 0 0 | 0 0 0 0000000 0 0 | 1 1 0 0000001 | 0 0 0 0000000
# T=65000 | 1   | 1 0 0 0000003 0 0 | 0 0 0 0000000 0 0 | 1 0 0 0000002 | 0 0 0 0000000
# T=75000 | 1   | 1 0 1 0000004 0 0 | 0 0 0 0000000 0 0 | 1 0 0 0000003 | 0 0 0 0000000
# T=85000 | 1   | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 0 | 1 0 1 0000004 | 0 0 0 0000000
# ==input port 1 to output port 0=====
# T=95000 | 1   | 0 0 0 0000000 0 0 | 1 1 0 0000005 0 1 | 0 0 0 0000000 | 0 0 0 0000000
# T=105000 | 1   | 0 0 0 0000000 0 0 | 1 0 0 0000006 0 1 | 1 1 0 0000005 | 0 0 0 0000000
# T=115000 | 1   | 0 0 0 0000000 0 0 | 1 0 0 0000007 0 1 | 1 0 0 0000006 | 0 0 0 0000000
# T=125000 | 1   | 0 0 0 0000000 0 0 | 1 0 1 0000008 0 1 | 1 0 0 0000007 | 0 0 0 0000000
# T=135000 | 1   | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 1 | 1 0 1 0000008 | 0 0 0 0000000
# ==input port 0 to output port 1=====
# T=145000 | 1   | 1 1 0 0000001 1 0 | 0 0 0 0000000 0 1 | 0 0 0 0000000 | 0 0 0 0000000
# T=155000 | 1   | 1 0 0 0000002 1 0 | 0 0 0 0000000 0 1 | 0 0 0 0000000 | 1 1 0 0000001
# T=165000 | 1   | 1 0 1 0000004 1 0 | 0 0 0 0000000 0 1 | 0 0 0 0000000 | 1 0 0 0000002
# T=175000 | 1   | 0 0 0 0000000 1 0 | 0 0 0 0000000 0 1 | 0 0 0 0000000 | 1 0 1 0000004
# ==input port 1 to output port 1=====
# T=185000 | 1   | 0 0 0 0000000 1 0 | 1 1 0 0000001 1 1 | 0 0 0 0000000 | 0 0 0 0000000
# T=195000 | 1   | 0 0 0 0000000 1 0 | 1 0 0 0000002 1 1 | 0 0 0 0000000 | 0 0 0 0000000
# T=205000 | 1   | 0 0 0 0000000 1 0 | 1 0 1 0000004 1 1 | 0 0 0 0000000 | 0 0 0 0000000
# T=215000 | 1   | 0 0 0 0000000 1 0 | 0 0 0 0000000 1 1 | 0 0 0 0000000 | 0 0 0 0000000
# ==input port 1 to output port 1, input port 0 to output port 0=====
# T=225000 | 1   | 1 1 0 0000001 0 0 | 1 1 0 0000001 1 1 | 0 0 0 0000000 | 0 0 0 0000000
# T=235000 | 1   | 1 0 0 0000001 0 0 | 1 1 0 0000002 1 1 | 1 1 0 0000001 | 0 0 0 0000000
# T=245000 | 1   | 1 0 0 0000001 0 0 | 1 0 0 0000003 1 1 | 1 0 0 0000001 | 0 0 0 0000000
# T=255000 | 1   | 1 0 1 0000001 0 0 | 1 0 1 0000004 1 1 | 1 0 0 0000001 | 0 0 0 0000000
# T=265000 | 1   | 0 0 0 0000000 0 0 | 0 0 0 0000000 1 1 | 1 0 1 0000001 | 0 0 0 0000000
# ==input port 1 to output port 0, input port 0 to output port 0=====
# T=275000 | 1   | 1 1 0 0000001 0 0 | 1 1 0 0000001 0 1 | 0 0 0 0000000 | 0 0 0 0000000
# T=285000 | 1   | 1 0 0 0000001 0 0 | 1 1 0 0000002 0 1 | 1 1 0 0000001 | 0 0 0 0000000
# T=295000 | 1   | 1 0 0 0000001 0 0 | 1 0 0 0000003 0 1 | 1 0 0 0000001 | 0 0 0 0000000
# T=305000 | 1   | 1 0 1 0000001 0 0 | 1 0 1 0000004 0 1 | 1 0 0 0000001 | 0 0 0 0000000
# T=315000 | 1   | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 1 | 1 0 1 0000001 | 0 0 0 0000000

```

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

```

After Bug Fix:

```

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

```

Transcript

```

# ==input port 1 to output port 0=====
# T=95000 | 1 | 0 0 0 0000000 0 0 | 1 1 0 0000005 0 1 | 0 0 0 0000000 | 0 0 0 0000000
# T=105000 | 1 | 0 0 0 0000000 0 0 | 1 0 0 0000006 0 1 | 1 1 0 0000005 | 0 0 0 0000000
# T=115000 | 1 | 0 0 0 0000000 0 0 | 1 0 0 0000007 0 1 | 1 0 0 0000006 | 0 0 0 0000000
# T=125000 | 1 | 0 0 0 0000000 0 0 | 1 0 1 0000008 0 1 | 1 0 0 0000007 | 0 0 0 0000000
# T=135000 | 1 | 0 0 0 0000000 0 0 | 0 0 0 0000008 0 1 | 1 0 1 0000008 | 0 0 0 0000000
# ==input port 0 to output port 1=====
# T=145000 | 1 | 1 1 0 0000001 1 0 | 0 0 0 0000000 0 1 | 0 0 0 0000000 | 0 0 0 0000000
# T=155000 | 1 | 1 0 0 0000002 1 0 | 0 0 0 0000000 0 1 | 0 0 0 0000000 | 1 1 0 0000001
# T=165000 | 1 | 1 0 1 0000004 1 0 | 0 0 0 0000000 0 1 | 0 0 0 0000000 | 1 0 0 0000002
# T=175000 | 1 | 0 0 0 0000000 1 0 | 0 0 0 0000000 0 1 | 0 0 0 0000000 | 1 0 1 0000004
# ==input port 1 to output port 1=====
# T=185000 | 1 | 0 0 0 0000000 1 0 | 1 1 0 0000001 1 1 | 0 0 0 0000000 | 0 0 0 0000000
# T=195000 | 1 | 0 0 0 0000000 1 0 | 1 0 0 0000002 1 1 | 0 0 0 0000000 | 1 1 0 0000001
# T=205000 | 1 | 0 0 0 0000000 1 0 | 1 0 1 0000004 1 1 | 0 0 0 0000000 | 1 0 0 0000002
# T=215000 | 1 | 0 0 0 0000000 1 0 | 0 0 0 0000000 1 1 | 0 0 0 0000000 | 1 0 1 0000004
# ==input port 1 to output port 0, input port 0 to output port 1=====
# T=225000 | 1 | 1 1 0 0000001 0 0 | 1 1 0 0000001 1 1 | 0 0 0 0000000 | 0 0 0 0000000
# T=235000 | 1 | 1 0 0 0000001 0 0 | 1 1 0 0000002 1 1 | 1 1 0 0000001 | 1 1 0 0000001
# T=245000 | 1 | 1 0 0 0000001 0 0 | 1 0 0 0000003 1 1 | 1 0 0 0000001 | 1 1 0 0000002
# T=255000 | 1 | 1 0 1 0000001 0 0 | 1 0 1 0000004 1 1 | 1 0 0 0000001 | 1 0 0 0000003
# T=265000 | 1 | 0 0 0 0000000 0 0 | 0 0 0 0000000 1 1 | 1 0 1 0000001 | 1 0 1 0000004
# ==input port 1 to output port 0, input port 0 to output port 0=====
# T=275000 | 1 | 1 1 0 0000001 0 0 | 1 1 0 0000001 0 1 | 0 0 0 0000000 | 0 0 0 0000000
# T=285000 | 1 | 1 0 0 0000001 0 0 | 1 1 0 0000002 0 1 | 1 1 0 0000001 | 0 0 0 0000000
# T=295000 | 1 | 1 0 0 0000001 0 0 | 1 0 0 0000003 0 1 | 1 1 0 0000002 | 0 0 0 0000000
# T=305000 | 1 | 1 0 1 0000001 0 0 | 1 0 1 0000004 0 1 | 1 0 0 0000003 | 0 0 0 0000000
# T=315000 | 1 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 1 | 1 0 1 0000004 | 0 0 0 0000000
# ==input port 1 to output port 1, input port 0 to output port 1=====
# T=325000 | 1 | 1 1 0 0000001 1 0 | 1 1 0 0000001 1 1 | 1 1 0 0000001 | 0 0 0 0000000
# T=335000 | 1 | 1 0 0 0000001 1 0 | 1 1 0 0000002 1 1 | 1 0 0 0000001 | 1 1 0 0000001
# T=345000 | 1 | 1 0 0 0000001 1 0 | 1 0 0 0000003 1 1 | 1 1 0 0000001 | 1 1 0 0000002
# T=355000 | 1 | 1 0 1 0000001 1 0 | 1 0 1 0000004 1 1 | 1 0 0 0000001 | 1 0 0 0000003
# T=365000 | 1 | 0 0 0 0000000 1 0 | 1 1 0 0000001 1 1 | 1 1 0 0000001 | 1 0 1 0000004
# T=375000 | 1 | 0 0 0 0000000 1 0 | 1 0 1 0000004 1 1 | 1 0 0 0000001 | 1 1 0 0000001
# T=385000 | 1 | 0 0 0 0000000 1 0 | 1 0 0 0000004 1 1 | 0 0 0 0000000 | 1 0 1 0000004
# T=395000 | 1 | 0 0 0 0000000 1 0 | 1 0 1 0000004 1 1 | 0 0 0 0000000 | 1 0 0 0000004
# =====
# T=405000 | 1 | 0 0 0 0000000 1 0 | 0 0 0 0000000 1 1 | 0 0 0 0000000 | 1 0 1 0000004
# T=415000 | 1 | 0 0 0 0000000 1 0 | 0 0 0 0000000 1 1 | 0 0 0 0000000 | 0 0 0 0000000
# T=425000 | 1 | 0 0 0 0000000 1 0 | 0 0 0 0000000 1 1 | 0 0 0 0000000 | 0 0 0 0000000
# T=435000 | 1 | 0 0 0 0000000 1 0 | 0 0 0 0000000 1 1 | 0 0 0 0000000 | 0 0 0 0000000

```

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 0000000 0 0 | 1 1 0 00000005 0 1 | 0 0 0 0000000 | 0 0 0 0000000

# T=105000 | 1 | 0 0 0 0000000 0 0 | 1 0 0 00000006 0 1 | 1 1 0 00000005 | 0 0 0 0000000

# T=115000 | 1 | 0 0 0 0000000 0 0 | 1 0 0 00000007 0 1 | 1 0 0 00000006 | 0 0 0 0000000

# T=125000 | 1 | 0 0 0 0000000 0 0 | 1 0 1 00000008 0 1 | 1 0 0 00000007 | 0 0 0 0000000

# T=135000 | 1 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 1 | 1 0 1 00000008 | 0 0 0 0000000

# ==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 0000000 | 0 0 0 0000000

# T=155000 | 1 | 1 0 0 00000002 1 0 | 0 0 0 00000000 0 1 | 0 0 0 0000000 | 1 1 0 00000001

# T=165000 | 1 | 1 0 1 00000004 1 0 | 0 0 0 00000000 0 1 | 0 0 0 0000000 | 1 0 0 00000002

# T=175000 | 1 | 0 0 0 00000000 1 0 | 0 0 0 00000000 0 1 | 0 0 0 0000000 | 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 0000000 | 0 0 0 0000000

# T=195000 | 1 | 0 0 0 00000000 1 0 | 1 0 0 00000002 1 1 | 0 0 0 0000000 | 1 1 0 00000001

# T=205000 | 1 | 0 0 0 00000000 1 0 | 1 0 1 00000004 1 1 | 0 0 0 0000000 | 1 0 0 00000002

# T=215000 | 1 | 0 0 0 00000000 1 0 | 0 0 0 00000000 1 1 | 0 0 0 0000000 | 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 0000000 | 0 0 0 0000000

# 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 1=====

# T=275000 | 1 | 1 1 0 00000001 0 0 | 1 1 0 00000001 0 1 | 0 0 0 0000000 | 0 0 0 0000000

# T=285000 | 1 | 1 0 0 00000001 0 0 | 1 1 0 00000002 0 1 | 1 1 0 00000001 | 0 0 0 0000000

# T=295000 | 1 | 1 0 0 00000001 0 0 | 1 0 0 00000003 0 1 | 1 1 0 00000002 | 0 0 0 0000000

# T=305000 | 1 | 1 0 1 00000001 0 0 | 1 0 1 00000004 0 1 | 1 0 0 00000003 | 0 0 0 0000000

# T=315000 | 1 | 0 0 0 00000000 0 0 | 0 0 0 00000000 0 1 | 1 0 1 00000004 | 0 0 0 0000000

# ==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 0000000

# 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 00000000 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 00000001 | 1 0 1 00000004

# T=395000 | 1 | 0 0 0 00000000 1 0 | 1 0 1 00000004 1 1 | 0 0 0 00000001 | 1 0 0 00000004

# =====

# T=405000 | 1 | 0 0 0 00000000 1 0 | 0 0 0 00000000 1 1 | 0 0 0 0000000 | 1 0 1 00000004

# T=415000 | 1 | 0 0 0 00000000 1 0 | 0 0 0 00000000 1 1 | 0 0 0 0000000 | 0 0 0 0000000

# T=425000 | 1 | 0 0 0 00000000 1 0 | 0 0 0 00000000 1 1 | 0 0 0 0000000 | 0 0 0 0000000

# T=435000 | 1 | 0 0 0 00000000 1 0 | 0 0 0 00000000 1 1 | 0 0 0 0000000 | 0 0 0 0000000

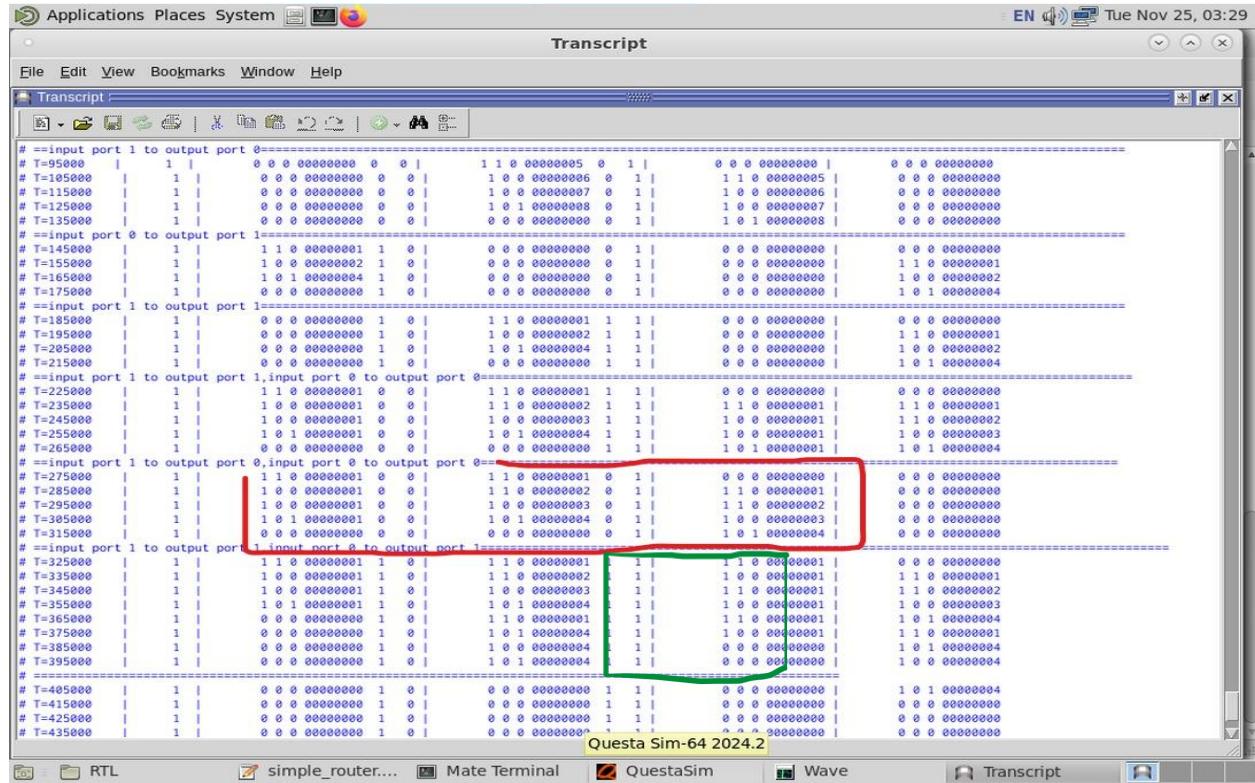
Applications Places System Transcript Tue Nov 25, 03:29

File Edit View Bookmarks Window Help

Transcript

```
# ==input port 1 to output port 0
# T=95000 | 1 |
# T=105000 | 1 |
# T=115000 | 1 |
# T=125000 | 1 |
# T=125000 | 1 |
# ==input port 0 to output port 1
# T=145000 | 1 |
# T=155000 | 1 |
# T=165000 | 1 |
# T=175000 | 1 |
# ==input port 1 to output port 1
# T=185000 | 1 |
# T=195000 | 1 |
# T=205000 | 1 |
# T=215000 | 1 |
# ==input port 1 to output port 1, input port 0 to output port 0=
# T=225000 | 1 |
# T=235000 | 1 |
# T=245000 | 1 |
# T=255000 | 1 |
# T=265000 | 1 |
# ==input port 1 to output port 0, input port 0 to output port 0=
# T=275000 | 1 |
# T=285000 | 1 |
# T=295000 | 1 |
# T=305000 | 1 |
# T=315000 | 1 |
# ==input port 1 to output port 1, input port 0 to output port 1
# T=325000 | 1 |
# T=335000 | 1 |
# T=345000 | 1 |
# T=355000 | 1 |
# T=365000 | 1 |
# T=375000 | 1 |
# T=385000 | 1 |
# T=395000 | 1 |
#
# T=405000 | 1 |
# T=415000 | 1 |
# T=425000 | 1 |
# T=435000 | 1 |
#
# Questa Sim-64 2024.2
```

## Fifo Issue:



```

# ==input port 1 to output port 0=====
# T=95000 | 1 | 0 0 0 0000000 0 0 | 1 1 0 0000005 0 1 | 0 0 0 0000000 | 0 0 0 0000000
# T=105000 | 1 | 0 0 0 0000000 0 0 | 1 0 0 0000006 0 1 | 1 1 0 0000005 | 0 0 0 0000000
# T=115000 | 1 | 0 0 0 0000000 0 0 | 1 0 0 0000007 0 1 | 1 0 0 0000006 | 0 0 0 0000000
# T=125000 | 1 | 0 0 0 0000000 0 0 | 1 0 1 0000008 0 1 | 1 0 0 0000007 | 0 0 0 0000000
# T=135000 | 1 | 0 0 0 0000000 0 0 | 0 0 0 0000008 0 1 | 1 0 1 0000008 | 0 0 0 0000000
# ==input port 0 to output port 1=====
# T=145000 | 1 | 1 1 0 0000001 1 0 | 0 0 0 0000000 0 1 | 0 0 0 0000000 | 0 0 0 0000000
# T=155000 | 1 | 1 0 0 0000002 1 0 | 0 0 0 0000000 0 1 | 0 0 0 0000000 | 1 1 0 0000001
# T=165000 | 1 | 1 0 1 0000004 1 0 | 0 0 0 0000000 0 1 | 0 0 0 0000000 | 1 0 0 0000002
# T=175000 | 1 | 0 0 0 0000000 1 0 | 0 0 0 0000000 0 1 | 0 0 0 0000000 | 1 0 1 0000004
# ==input port 1 to output port 1=====
# T=185000 | 1 | 0 0 0 0000001 1 0 | 1 1 0 0000001 1 1 | 0 0 0 0000000 | 0 0 0 0000000
# T=195000 | 1 | 0 0 0 0000000 1 0 | 1 0 0 0000002 1 1 | 0 0 0 0000000 | 1 1 0 0000001
# T=205000 | 1 | 0 0 0 0000000 1 0 | 1 0 1 0000004 1 1 | 0 0 0 0000000 | 1 0 0 0000002
# T=215000 | 1 | 0 0 0 0000000 1 0 | 0 0 0 0000000 1 1 | 0 0 0 0000000 | 1 0 1 0000004
# ==input port 1 to output port 1, input port 0 to output port 0=====
# T=225000 | 1 | 1 1 0 0000001 0 0 | 1 1 0 0000001 1 1 | 0 0 0 0000000 | 0 0 0 0000000
# T=235000 | 1 | 1 0 0 0000001 0 0 | 1 0 0 0000002 1 1 | 1 1 0 0000001 | 1 1 0 0000001
# T=245000 | 1 | 1 0 0 0000001 0 0 | 1 0 0 0000003 1 1 | 1 0 0 0000001 | 1 1 0 0000002
# T=255000 | 1 | 1 0 1 0000001 0 0 | 1 0 1 0000004 1 1 | 1 0 0 0000001 | 1 0 0 0000003
# T=265000 | 1 | 0 0 0 0000000 0 0 | 0 0 0 0000000 1 1 | 1 0 1 0000001 | 1 0 1 0000004
# ==input port 1 to output port 0, input port 0 to output port 0=====
# T=275000 | 1 | 1 1 0 0000001 0 0 | 1 1 0 0000001 0 1 | 0 0 0 0000000 | 0 0 0 0000000
# T=285000 | 1 | 1 0 0 0000001 0 0 | 1 0 0 0000002 0 1 | 1 1 0 0000001 | 0 0 0 0000000
# T=295000 | 1 | 1 0 0 0000001 0 0 | 1 0 0 0000003 0 1 | 1 1 0 0000002 | 0 0 0 0000000
# T=305000 | 1 | 1 0 1 0000001 0 0 | 1 0 1 0000004 0 1 | 1 0 0 0000003 | 0 0 0 0000000
# T=315000 | 1 | 0 0 0 0000000 0 0 | 0 0 0 0000000 0 1 | 1 0 1 0000004 | 0 0 0 0000000
# ==input port 1 to output port 1, input port 0 to output port 1=====
# T=325000 | 1 | 1 1 0 0000001 1 0 | 1 1 0 0000001 1 1 | 1 1 0 0000001 | 0 0 0 0000000
# T=335000 | 1 | 1 0 0 0000001 1 0 | 1 0 0 0000002 1 1 | 1 0 0 0000001 | 1 1 0 0000001
# T=345000 | 1 | 1 0 0 0000001 1 0 | 1 0 0 0000003 1 1 | 1 0 0 0000001 | 1 1 0 0000002
# T=355000 | 1 | 1 0 1 0000001 1 0 | 1 0 1 0000004 1 1 | 1 0 0 0000001 | 1 0 0 0000003
# T=365000 | 1 | 0 0 0 0000000 1 0 | 1 0 0 0000001 1 1 | 1 0 0 0000001 | 1 0 1 0000004
# T=375000 | 1 | 0 0 0 0000000 1 0 | 1 0 1 0000004 1 1 | 1 0 0 0000001 | 1 1 0 0000001
# T=385000 | 1 | 0 0 0 0000000 1 0 | 1 0 0 0000004 1 1 | 0 0 0 0000000 | 1 0 1 0000004
# T=395000 | 1 | 0 0 0 0000000 1 0 | 1 0 1 0000004 1 1 | 0 0 0 0000000 | 1 0 0 0000000
# =====
# T=405000 | 1 | 0 0 0 0000000 1 0 | 0 0 0 0000000 1 1 | 0 0 0 0000000 | 1 0 1 0000004
# T=415000 | 1 | 0 0 0 0000000 1 0 | 0 0 0 0000000 1 1 | 0 0 0 0000000 | 0 0 0 0000000
# T=425000 | 1 | 0 0 0 0000000 1 0 | 0 0 0 0000000 1 1 | 0 0 0 0000000 | 0 0 0 0000000
# T=435000 | 1 | 0 0 0 0000000 1 0 | 0 0 0 0000000 1 1 | 0 0 0 0000000 | 0 0 0 0000000

```

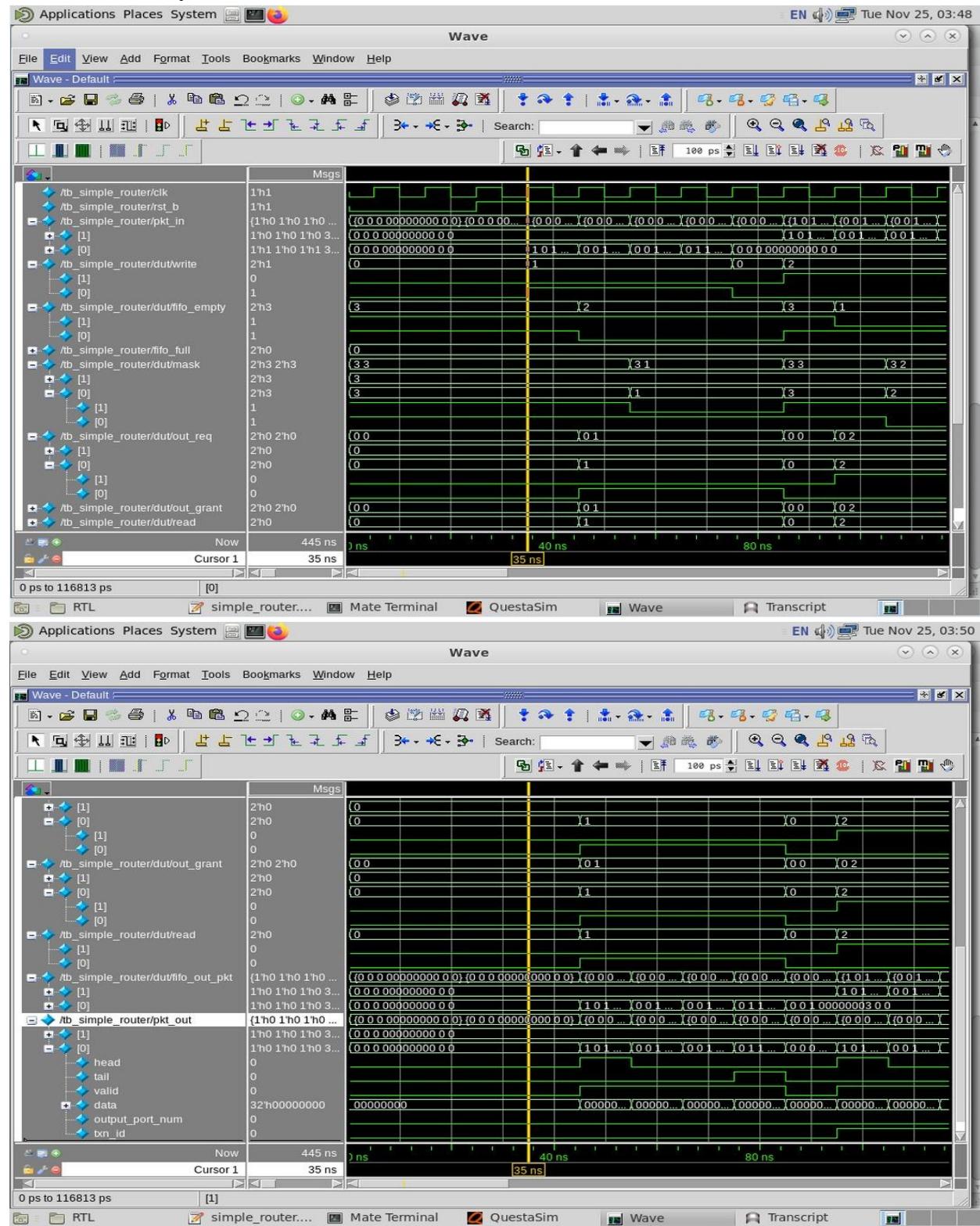
The green-marked area in the screenshot shows that the previously stored head and tail flits from input 0 to FIFO 0 are being sent again through output 0. This happens because the data in FIFO 0 is not being removed after it is read. Since the FIFO cell is not cleared, the same data keeps repeating, 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(output0 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]

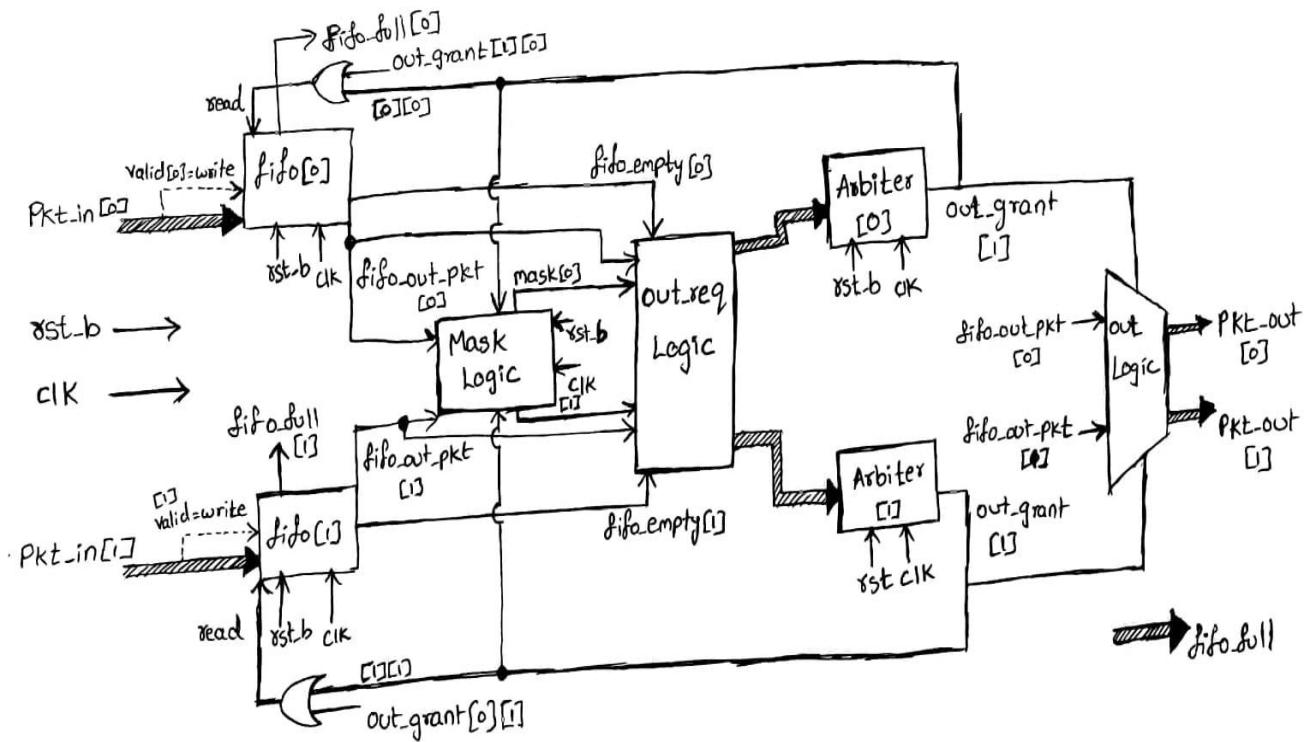
- When the head FLIT is received, the write signal for FIFO[0] goes high and FIFO\_EMPTY goes low. This indicates that FIFO[0] now contains data. As a result, the request signal OUT\_REQ[0][0] goes high because output\_port\_num = 0 and mask[0][0] is also high.
- Once OUT\_GRANT[0][0] becomes high, the 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] == 01	mask[0] == 11	
9			mask[1] == 11	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	req to output_0		
11			gnt to input_0	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	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	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	pkt_out[0].tail		
15									

### 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]	I pkt_in[0].valid, I write[0] I pkt_in[1].valid, I 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'h2	pkt_in[0].data == 32'h1 pkt_in[1].data == 32'h4			
7			fifo_empty[0] fifo_empty[1]	I fifo_empty[0] I fifo_empty[1]	I fifo_empty[0] I fifo_empty[1]	I fifo_empty[0] I fifo_empty[1]	I fifo_empty[0] I 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		
16									

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]	I pkt_in[0].valid, I write[0] I pkt_in[1].valid, I 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]	I fifo_empty[0] I fifo_empty[1]	I fifo_empty[0] I fifo_empty[1]	I fifo_empty[0] I fifo_empty[1]	I fifo_empty[0] I fifo_empty[1]	I fifo_empty[0] I fifo_empty[1]
mask[0] == 11 mask[1] == 11		mask[0] == 11 mask[1] == 11	mask[0] == 10 mask[1] == 11	mask[0] == 10 mask[1] == 11	mask[0] == 10 mask[1] == 11	mask[0] == 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	
		I out0_gnt to input_0 out0_gnt to input_1	I out0_gnt to input_0 out0_gnt to input_1	I out0_gnt to input_0 out0_gnt to input_1	I out0_gnt to input_0 out0_gnt to input_1	
		I read_0 to input_0 read_1 to input_1	I read_0 to input_0 read_1 to input_1	I read_0 to input_0 read_1 to input_1	I 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:

The screenshot shows the VC Formal FPV tool interface. The top menu bar includes File, View, Source, OneTrace, Tools, Window, and Help. The toolbar contains icons for opening files, saving, running tasks, and other tools. The main window is divided into several panes:

- VCF:TaskList**: Shows a task named "FPV" with a progress bar at 24:00:00.
- VCF:GoalList**: Displays a table of assertions:
 

status (V)	depth	name	vacuity	witness	type	engine	elapsed_time
X 3	3	...sertions.P_MaskReset_O0_assert	✓ 2		assert	rf1	00:00:01
X 3	3	...sertions.P_MaskReset_O1_assert	✓ 2		assert	rf1	00:00:01
X 2	2	...sertions.P_Route_In0_Out1_assert	✓ 2		assert	rf1	00:00:01
X 2	2	...sertions.P_Route_In1_Out0_assert	✓ 2		assert	rf1	00:00:01

 A summary table below shows the overall status:
 

Task	Property Summary	Assertion	Cover	Constraint	Vacuity	Witness
FPV	Number of Properties Proven (P) / Covered (C) Falsified (F) / Uncoverable (U) Vacuous (V)	22 18 4 0	2 2 0 -	16 - - -	32 32 0 0	0 0 0 0

 A message at the bottom of the GoalList pane states: "Total Properties: 24 - passed[ 20 ] - failed[ 4 ] - disabled[ 0 ] ; Constraints Enabled: 16 ; Min depth: 2 ; Max depth: 3 ; Run Time: 00:00:01".
- VC Formal Console**: Displays log messages:
 

```

80     Use 'set_grid_usage' to maximize worker usage and improve performance, if there are sufficient compute resources to support more workers.
81     Each runtime license supports 12 workers.
82 [Info] LIC_RT_CHECKOUT: VC Formal run time license checkout. Base:1 FPV:1.
83 [Info] RETRIEVE LEARN DATA: Retrieved learned data from (local directory) simple_router_learn_dir.
84 [Info] BITLEVEL_MODEL_STATS: Generated model with 2255 gates, 74 inputs, 245 registers, 0 initial constraints, 15 constraints.
85 [Info] BITLEVEL_MODEL_STATS: Generated model with 2546 gates, 74 inputs, 265 registers, 0 initial constraints, 15 constraints.
      
```
- Bottom Status Bar**: Shows tabs for "run", "Mate Terminal", "router\_fifo.sv (~/V...", "Mate Terminal", "S <Verdi-Apex:nTrac...", and "S".

## Bug Identification:

After we run the given design

The screenshot shows two windows from a verification tool. The top window is 'VCFGoalList' with 'Targets: ALL'. It lists 16 assertions (rows 1-16) with columns: status, depth, name, vacuity, witness, type, engine, and elapsed time. Rows 1 and 2 have a red 'X' in the status column, indicating they failed. The bottom window is 'Analyzer' showing Verilog code with annotations. A red arrow points to line 2, which contains the assertion 'P\_MaskReset\_00'. Another red arrow points to line 6, where a condition is shown as 'disabled iff (!rst\_b) (((out\_grant[0][0] & fifo\_out\_pkt[0].tail) && fifo\_out\_pkt[1].valid) || ((out\_grant[0][1] & fifo\_out\_pkt[1].tail) && fifo\_out\_pkt[0].valid))'. A red bracket below the condition is labeled 'FALSE'.

status	depth	name	vacuity	witness	type	engine	elapsed time
1	3	...erions.P_MaskReset_00_assert	✓ <sup>2</sup>		assert	b8	00:00:02
2	3	...erions.P_MaskReset_01_assert	✓ <sup>2</sup>		assert	b8	00:00:02
3		...erions.P_OneHot_Grant0_assert			assert	t1	00:00:01
4		...erions.P_OneHot_Grant1_assert			assert	t1	00:00:01
5		...ritions.P_OutMatch0_From0_assert	✓ <sup>2</sup>		assert	rp1	00:00:02
6		...ritions.P_OutMatch0_From1_assert	✓ <sup>2</sup>		assert	rp1	00:00:02
7		...ritions.P_OutMatch1_From0_assert	✓ <sup>2</sup>		assert	rp1	00:00:02
8		...ritions.P_OutMatch1_From1_assert	✓ <sup>2</sup>		assert	rp1	00:00:02
9		...ritions.P_Read_from_fifo_0_assert	✓ <sup>2</sup>		assert	t1	00:00:01
10		...ritions.P_Read_from_fifo_1_assert	✓ <sup>2</sup>		assert	t1	00:00:01
11		...erions.P_Route_In0_Out0_assert	✓ <sup>2</sup>		assert	rp1	00:00:02
12	2	...erions.P_Route_In0_Out1_assert	✓ <sup>2</sup>		assert	rf1	00:00:01
13	2	...erions.P_Route_In1_Out0_assert	✓ <sup>2</sup>		assert	rf1	00:00:01
14		...erions.P_Route_In1_Out1_assert	✓ <sup>2</sup>		assert	rp1	00:00:02
15		...erions.P_Write_to_fifo_0_assert	✓ <sup>1</sup>		assert	t1	00:00:01
16		erations.D_Writer_to_Rfn_1_assert	✗ <sup>1</sup>		assert	t1	00:00:01

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.

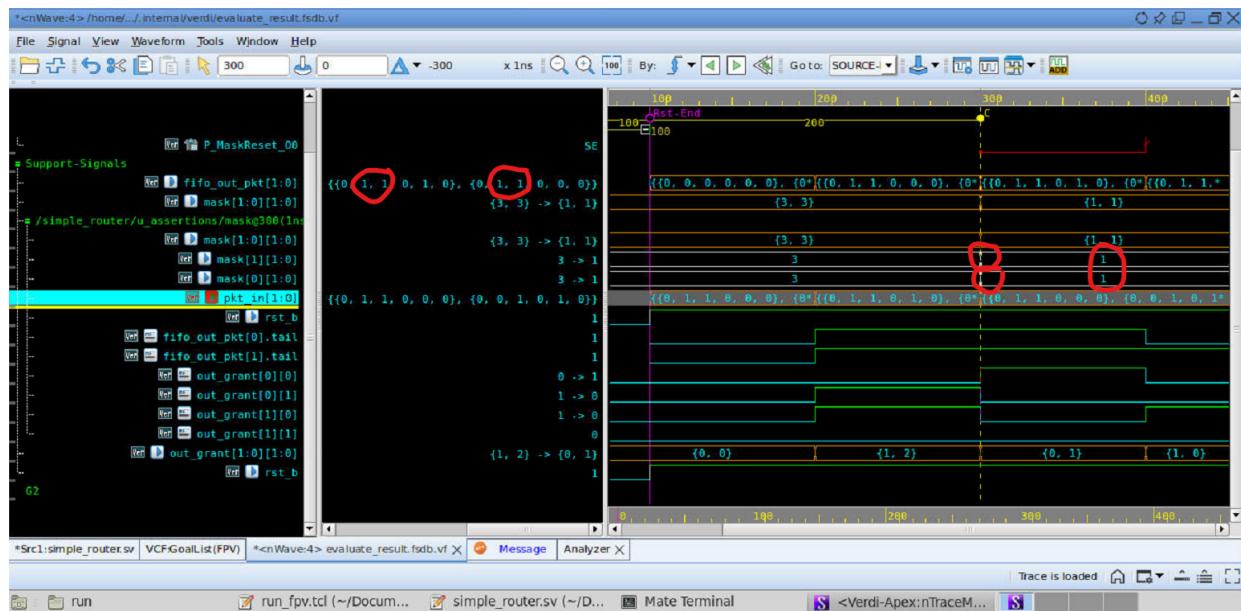
The screenshot shows a code editor with Verilog code. Line 55 starts with 'always\_ff @'(posedge clk or negedge rst\_b) begin. Line 56 contains an if block: 'if(!rst\_b) begin'. Line 61 is part of the if block: 'if((out\_grant[0][0] && fifo\_out\_pkt[0].tail) || (out\_grant[0][1] && fifo\_out\_pkt[1].tail)) begin'. Line 62 shows a condition: 'mask[0] <= 2'h1;'. Lines 63 and 64 show else branches: 'end' and 'else if (out\_grant[0][0]) begin'. Line 65 shows another condition: 'mask[0] <= 2'h1;'.

```

55 always_ff @(posedge clk or negedge rst_b) begin
56   if(!rst_b) begin
57     1
58   end
59   else begin
60
61     if((out_grant[0][0] && fifo_out_pkt[0].tail) || (out_grant[0][1] && fifo_out_pkt[1].tail)) begin
62       mask[0] <= 2'h1;
63     end
64     else if (out_grant[0][0]) begin
65       mask[0] <= 2'h1;

```

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] are 2'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
  end

```

### After Fix:

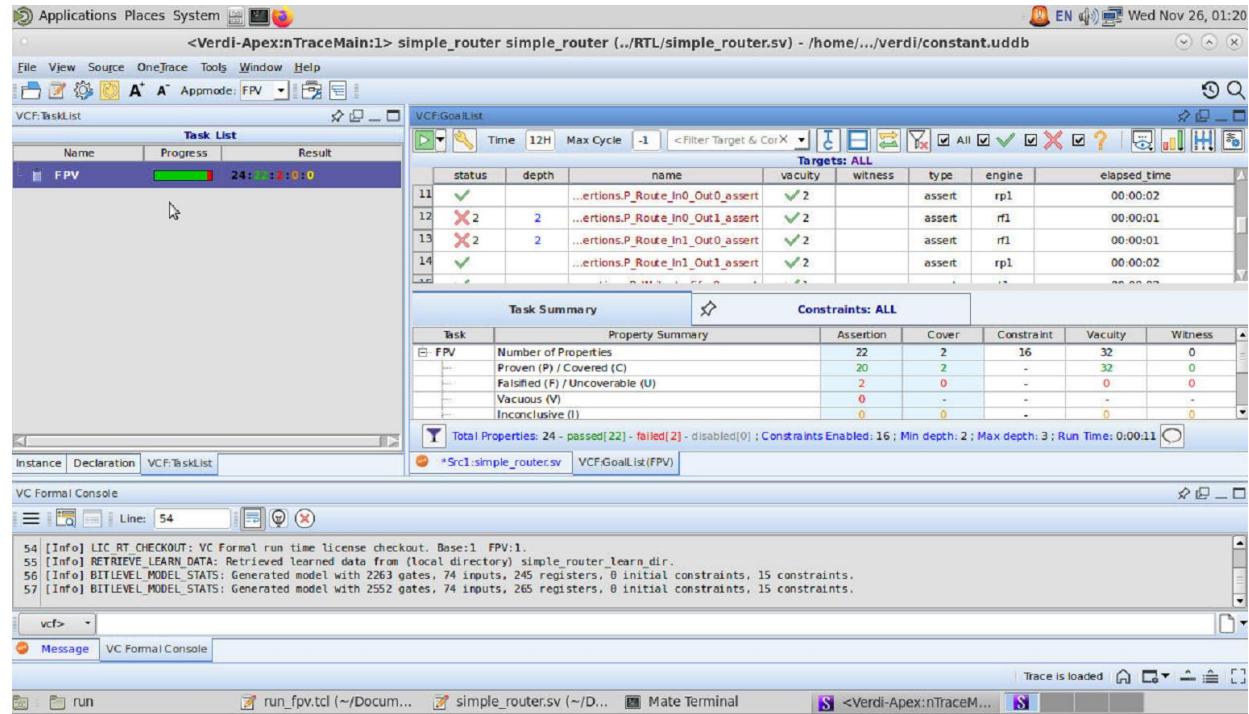
```

    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;
    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
  end

```

After running it again:



There are 2 remaining failed assertions.

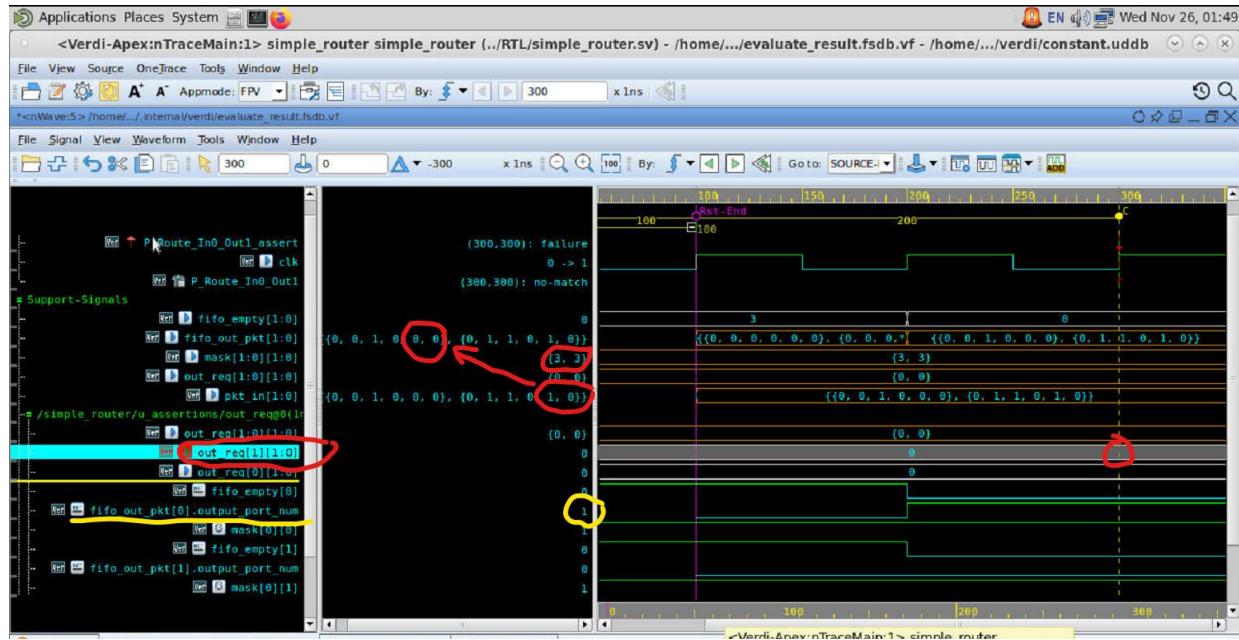
After double clicking on the x mark for failed assertion P\_Route\_In0\_Out1. I found out that out\_req[1][0] is low where it should be high. Let's check the wave form to check why it is 0.

The screenshot shows a waveform viewer with the following details:

- Code Snippet:**

```
P_Route_In0_Out1_assert: assert property(
  @(posedge clk) disable iff (!rst_b) ((( fifo_empty[0] ) && (fifo_out_pkt[0].output_port_num == 1'b1) ) && mask[1][0]) |-> out_req[1][0];
endproperty
```
- Waveform:** A timeline showing the signal "out\_req[1][0]" with two horizontal markers labeled "FALSE" at different points in time.

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)	S
0 out_req	{0, 0}	0	./RTL/route...ions.sva(11)	simple_router.u_assertions
+ assign out_req[0][0] = !fifo_empty[0] && mask[0][0] : 1'b0;		0	./RTL/simple...outer.sv(40)	simple_router
+ assign out_req[0][1] = !fifo_empty[1] && mask[0][1] : 1'b0;		0	./RTL/simple...outer.sv(41)	simple_router
+ assign out_req[1][0] = !fifo_empty[0] && mask[1][0] : 1'b0;		0	./RTL/simple...outer.sv(42)	simple_router
+ assign out_req[1][1] = !fifo_empty[1] && mask[1][1] : 1'b0;		0	./RTL/simple...outer.sv(43)	simple_router
0 fifo_out_pkt[0].output_port_num	NF	0	./RTL/simple...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
0 fifo_out_pkt[1].output_port_num	NF	0	./RTL/simple...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:

```

end : ROUTER_IMP_111
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;

```

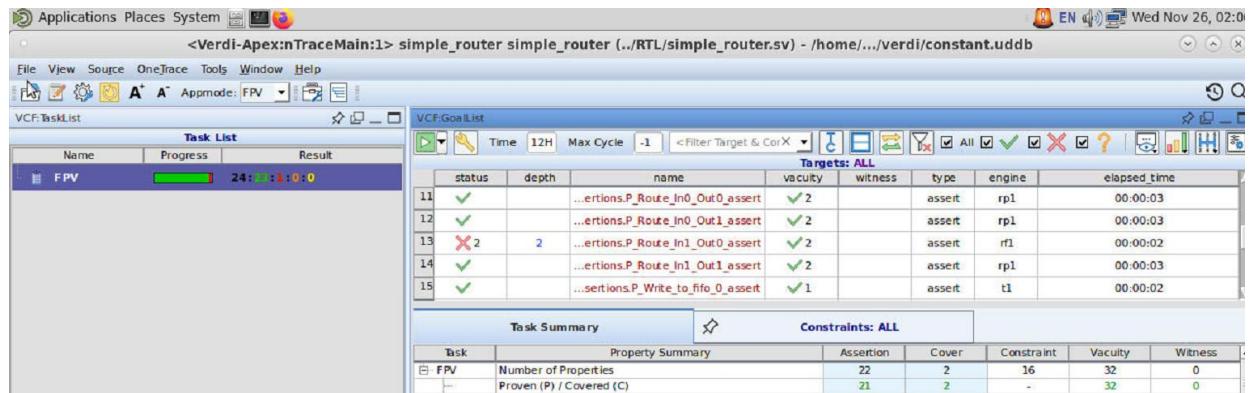
## After fixing the code:

```

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.

```

4

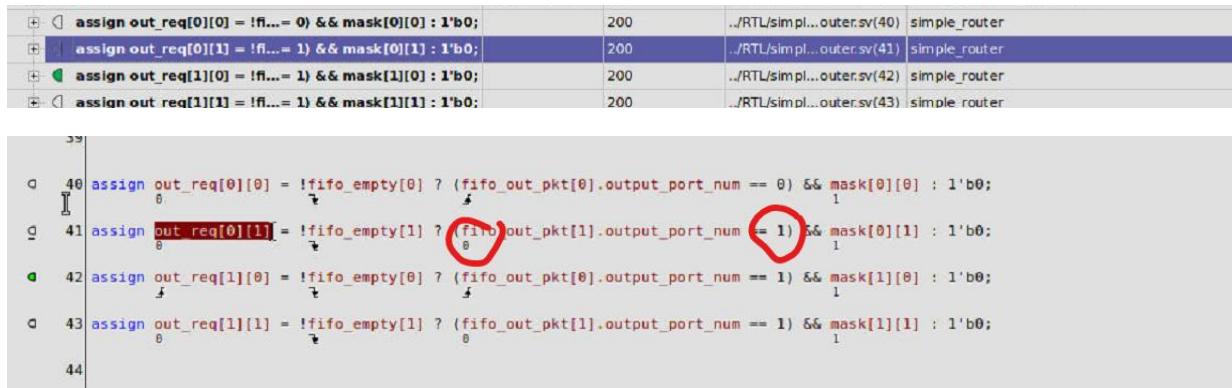
5 property P_Route_In1_Out0;
6   @posedge CLK disable iff (!rst_b) (((!fifo_empty[1]) && (fifo_out_pkt[1].output_port_num == 1'b0)) && mask[0][1]) |-> out_req[0][1];
7
8 endproperty
9

```

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:

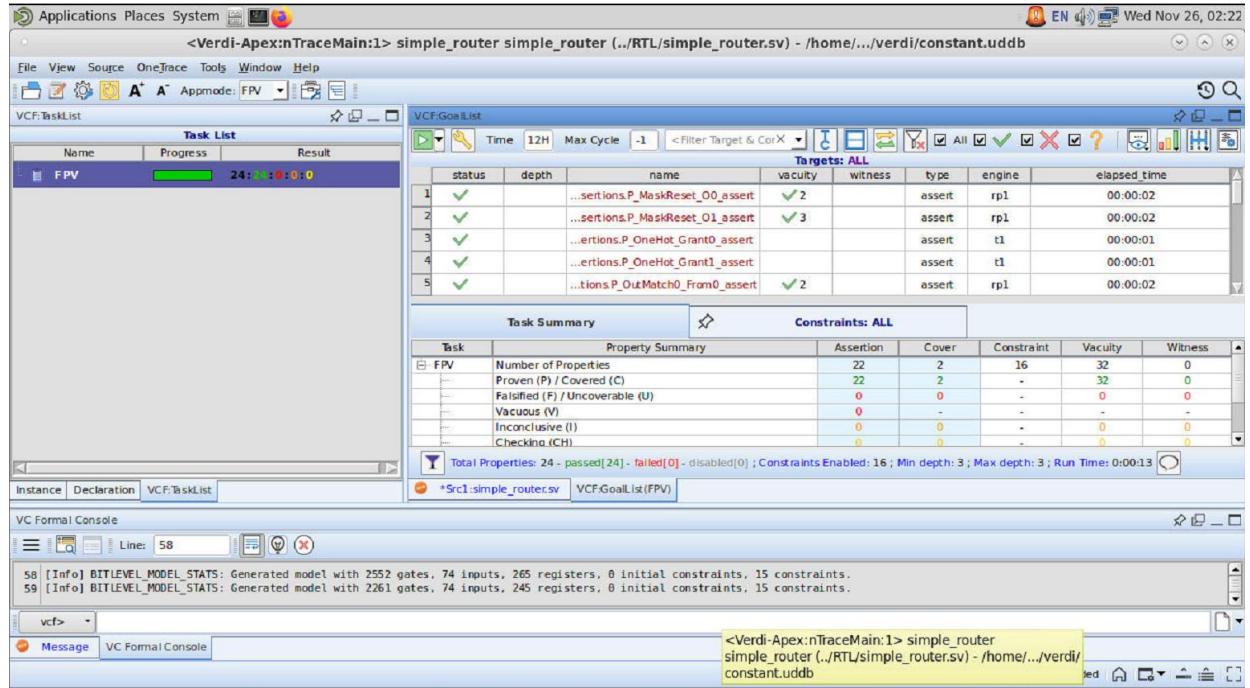
```
assign out_req[0][0] = !fifo_empty[0] ? (fifo_out_pkt[0].output_port_num == 0) && mask[0][0] : 1'b0;
assign out_req[0][1] = !fifo_empty[1] ? (fifo_out_pkt[1].output_port_num == 1) && mask[0][1] : 1'b0;
assign out_req[1][0] = !fifo_empty[0] ? (fifo_out_pkt[0].output_port_num == 1) && mask[1][0] :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

```
enabgenerate

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.udl

File View Source OneTrace Tools Window Help

VCF:TaskList VCF:GoalList

**Task List**

Name	Progress	Result
AEP	22:00:2	

**VCF:GoalList**

Targets: Success And Failure

status (V)	depth	name	type	location	expression	state_reg	state_name	state_val
1 ✓		..._priority_ptr_assign_i_plus_1b1_	arith_oflow	...t_arbiters.sv:28	... <= (i + 1'b1);			
2 ✓		...arb.bounds_check_0_out_grant_i	bounds_check	...t_arbiters.sv:26	out_grant[i]			
3 ✓		...bounds_check_1_out_req_index	bounds_check	...t_arbiters.sv:40	out_req[index]			

**Task Summary**

Task	Property Summary	Assertion	Cover	Constraint	Vacuity	Witness
AEP	Number of Properties Proven (P) / Covered (C) Falsified (F) / Uncoverable (U) Vacuous (V)	22 20 2 0	0 0 0 -	1 - - -	0 0 0 -	0 0 0 -

Total Properties: 22 - passed[20] - failed[2] - disabled[0] ; Constraints Enabled: 1 ; Min depth: 0 ; Max depth: 0 ; Run Time: 0:00:00

Instance Declaration VCF:TaskList

VC Formal Console

Rule View Tools Window Help

Line: 106

```

166 Use 'set_grid_usage' to maximize worker usage and improve performance, if there are sufficient compute resources to support more
workers.
167 Each runtime license supports 12 workers.
168 [Info] LIC RT CHECKOUT: VC Formal run time license checkout. Base:1 AEP:1.
169 [Info] RETRIEVE LEARN DATA: Retrieved learned data from (local directory) simple_router learn dir.
170 [Info] BITLEVEL MODEL STATS: Generated model with 48 gates, 0 inputs, 18 registers, 0 initial constraints, 0 constraints.
171 [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.udl

File View Source OneTrace Tools Window Help

VCF:GoalList

Targets: Success And Failure

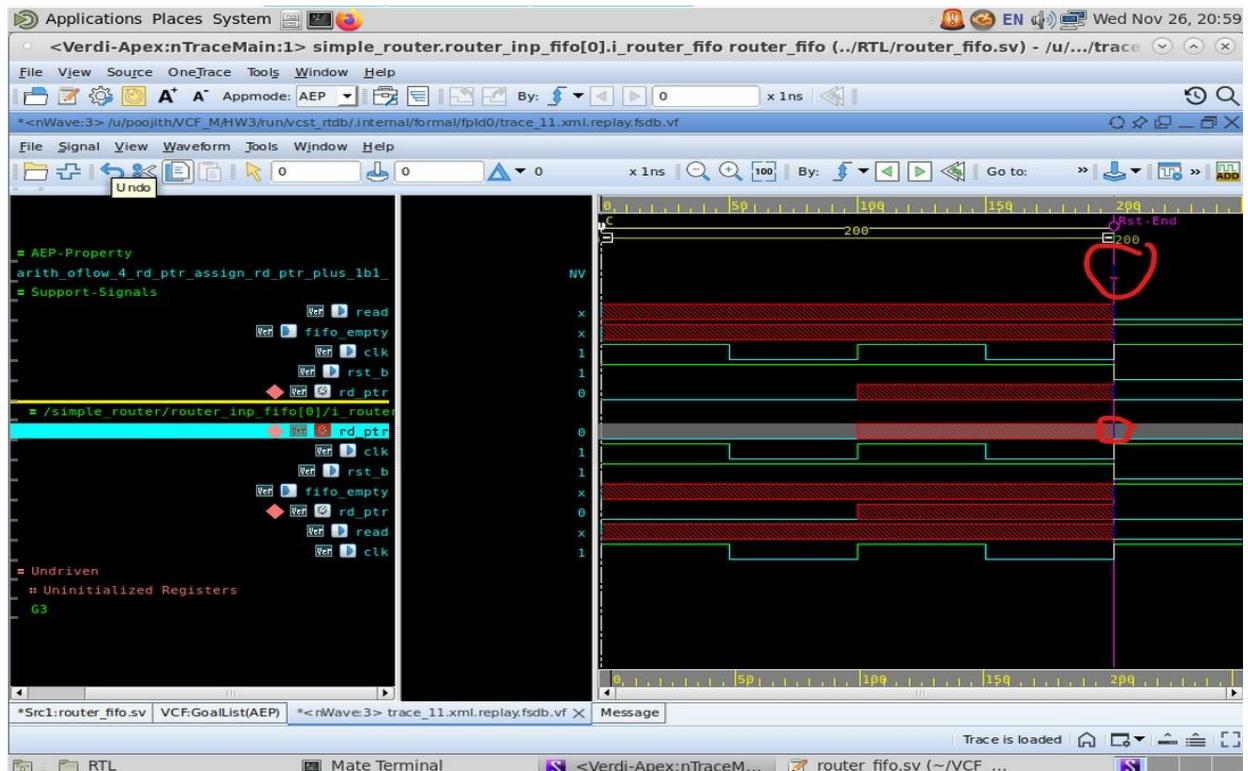
status (V)	depth	name	type	location	expression	state_reg	state_name	state_val	engine	elaps...time
11 ✓		...w_3_count_assign_count_plus_1_	arith_oflow	...uter_fifo.sv:44	... = (count + 1);				t1	00:00:02
12 ✗ 0	0	...rd_ptr_assign_rd_ptr_plus_1b1_	arith_oflow	...uter_fifo.sv:52	...rd_ptr + 1'b1;				rfl	00:00:02
13 ✓		...fifo.bounds_check_3_fifo_mem_i	bounds_check	...uter_fifo.sv:31	fifo_mem[i]				rfl	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]				rfl	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);				rfl	00:00:02
20 ✓		...fifo.bounds_check_3_fifo_mem_i	bounds_check	...uter_fifo.sv:31	fifo_mem[i]				rfl	00:00:02

**Task Summary**

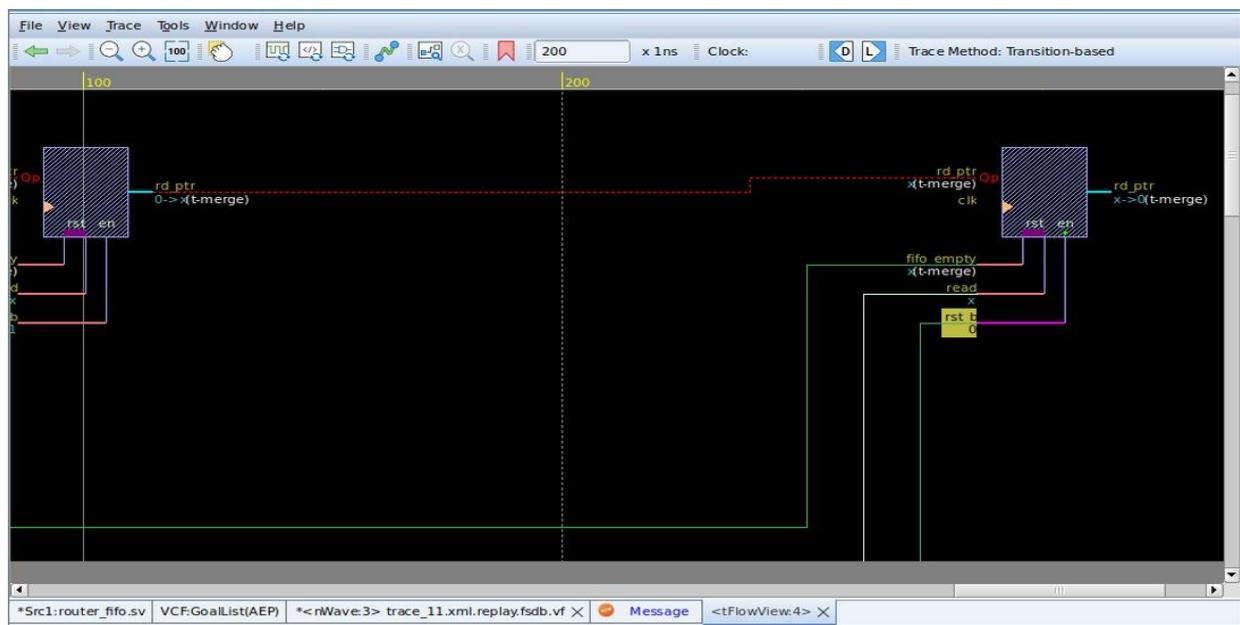
Task	Property Summary	Assertion	Cover	Constraint	Vacuity	Witness
AEP	Number of Properties Proven (P) / Covered (C) Falsified (F) / Uncoverable (U) Vacuous (V) Inconclusive (I) Checking (CH) Not run (N) Disabled	22 20 2 0 0 0 0 0	0 0 0 - 0 0 0 0	1 - - - 0 0 0 0	0 0 0 - 0 0 0 0	0 0 0 - 0 0 0 0

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 read pointer. So, failing to write takes much more complex sequence or a particular state of guarding signals.



```

42 // ASYNCHRONOUS RESET
43
44 if(~rst_b) begin
45   wr_ptr    <= '0;
46   x>>8
47
48   rd_ptr    <= '0;
49   x>>8
50
51   count <= '0;
52   X>>8
53
54   for(int i = 0; i < DEPTH; i++) begin
55     fifo_mem[i] <= '0;
56   end

```

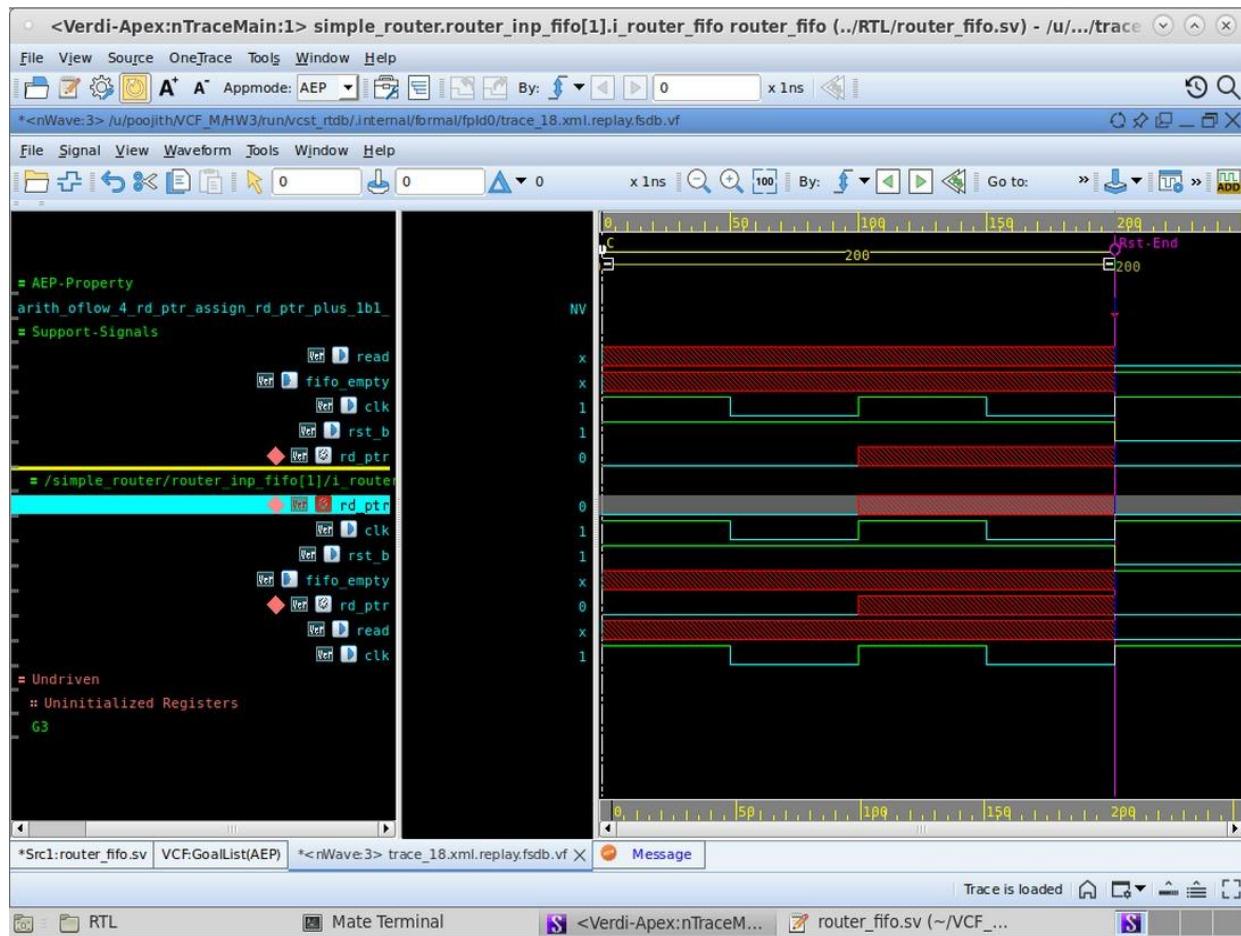
\*Src1:simple\_router.router\_inp\_fifo[0].i\_router\_fifo(/u/poojith/VCF\_M/HW3/RTL/router\_fifo.sv)

```

50
51 if(read && !fifo_empty) begin
52   rd_ptr[x>>8] <= rd_ptr[x>>8] + 1'b1;
53 end
54 end
55

```

## 2<sup>nd</sup> failed AEP CEX:



```

46  /*      if(read && !fifo_empty) begin
47
48      rd_ptr <= rd_ptr + 1'b1;
49
50
51  if(read && !fifo_empty) begin
52      rd_ptr[0] <= rd_ptr[0] + 1'b1;
53
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:

```

21 // Synchronous write and pointer update behavior
22 always_ff @ (posedge clk or negedge rst_b)
23 begin
24     // Asynchronous reset
25     if (~rst_b) begin
26         wr_ptr   <= '0;
27         rd_ptr   <= '0;
28         count    <= '0;
29         for (int i = 0; i < DEPTH; i++) begin
30             fifo_mem[i] <= '0;
31         end
32     end
33     else begin
34         if (write && !fifo_full) begin
35             fifo_mem[wr_ptr] <= fifo_in_pkt; // If we receive a write command, store the input packet to the slot pointed to
36             by write pointer
37             wr_ptr <= wr_ptr + 1'b1;           // Also increment the write pointer
38         end
39
40         if (read && !fifo_empty && !write) begin
41             count <= count - 1;
42         end
43         else if (write && !fifo_full && !read) begin
44             count <= count + 1;
45         end
46         if (read && !fifo_empty) begin //Improvement| After Jasper Gold FPV app and VC Formal AEP app.
47             rd_ptr <= rd_ptr + 1'b1;
48         end
49     end
50
51
52 end

```

## Result After Improvement:

The screenshot shows the Verdi-Apex interface with the following details:

- File Menu:** Applications, Places, System, File, View, Source, OneTrace, Tools, Window, Help.
- Toolbar:** Includes icons for Open, Save, Undo, Redo, Cut, Copy, Paste, Find, Replace, etc.
- Appmode:** AEP (top) and FXP (bottom).
- VCF:GoalList:** Task Summary and Constraints: ALL tables.
- Task Summary:**

Task	Property Summary	Assertion	Cover	Constraint	Vacuity	Witness
AEP	Number of Properties: 22 Proven (P) / Covered (C) Falsified (F) / Uncoverable (U) Vacuous (V) Inconclusive (I) Checking (CH) Not run (N) Disabled	22 22 0 0 0 0 0 0	0 0 0 0 0 0 0 0	1 - - - - - - -	0 0 0 0 0 0 0 -	0 0 0 0 0 0 0 -
- Constraints: ALL:**

Targets: Success And Failure
status (V)   depth   name   type   location   expression   state_reg   state_name   state_val   engine   elaps... time
1   ✓   ...priority_ptr_assign_i_plus_1bl_   arith_oflow   ...t_arbiter.sv:28   ... <= (i + 1'b1);   -   -   -   t1   00:00:02
2   ✓   ...arb_bounds_check_0_out_grant_i   bounds_check   ...t_arbiter.sv:26   out_grant[i]   -   -   -   t1   00:00:02
3   ✓   ...bounds_check_1_out_req_index   bounds_check   ...t_arbiter.sv:40   out_req[index]   -   -   -   rf1   00:00:01
4   ✓   ...ounds_check_2_out_grant_index   bounds_check   ...t_arbiter.sv:42   out_grant[index]   -   -   -   rf1   00:00:01
5   ✓   ...priority_ptr_assign_i_plus_1bl_   arith_oflow   ...t_arbiter.sv:28   ... <= (i + 1'b1);   -   -   -   t1   00:00:02
6   ✓   ...arb_bounds_check_0_out_grant_i   bounds_check   ...t_arbiter.sv:26   out_grant[i]   -   -   -   t1   00:00:02
7   ✓   ...bounds_check_1_out_req_index   bounds_check   ...t_arbiter.sv:40   out_req[index]   -   -   -   rf1   00:00:01
8   ✓   ...ounds_check_2_out_grant_index   bounds_check   ...t_arbiter.sv:42   out_grant[index]   -   -   -   rf1   00:00:01
9   ✓   ...wr_ptr_assign_wr_ptr_plus_1bl_   arith_oflow   ...uter_fifo.sv:37   ...r_ptr + 1'b1);   -   -   -   t1   00:00:02
- Bottom Status Bar:** Total Properties: 22 - passed[22] - failed[0] - disabled[0]; Constraints Enabled: 1; Run Time: 0:00:10.

## 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):

The screenshot shows the Verdi-Apex interface with the following details:

- File Menu:** Applications, Places, System, File, View, Source, OneTrace, Tools, Window, Help.
- Toolbar:** Includes icons for Open, Save, Undo, Redo, Cut, Copy, Paste, Find, Replace, etc.
- Appmode:** FXP (top) and VCF (bottom).
- VCF:TaskList:** Task List table.
- Task List:**

Name	Progress	Result
FXP	2/2 100% 0/0	Passed
- VCF:GoalList:** Task Summary and Constraints: ALL tables.
- Task Summary:**

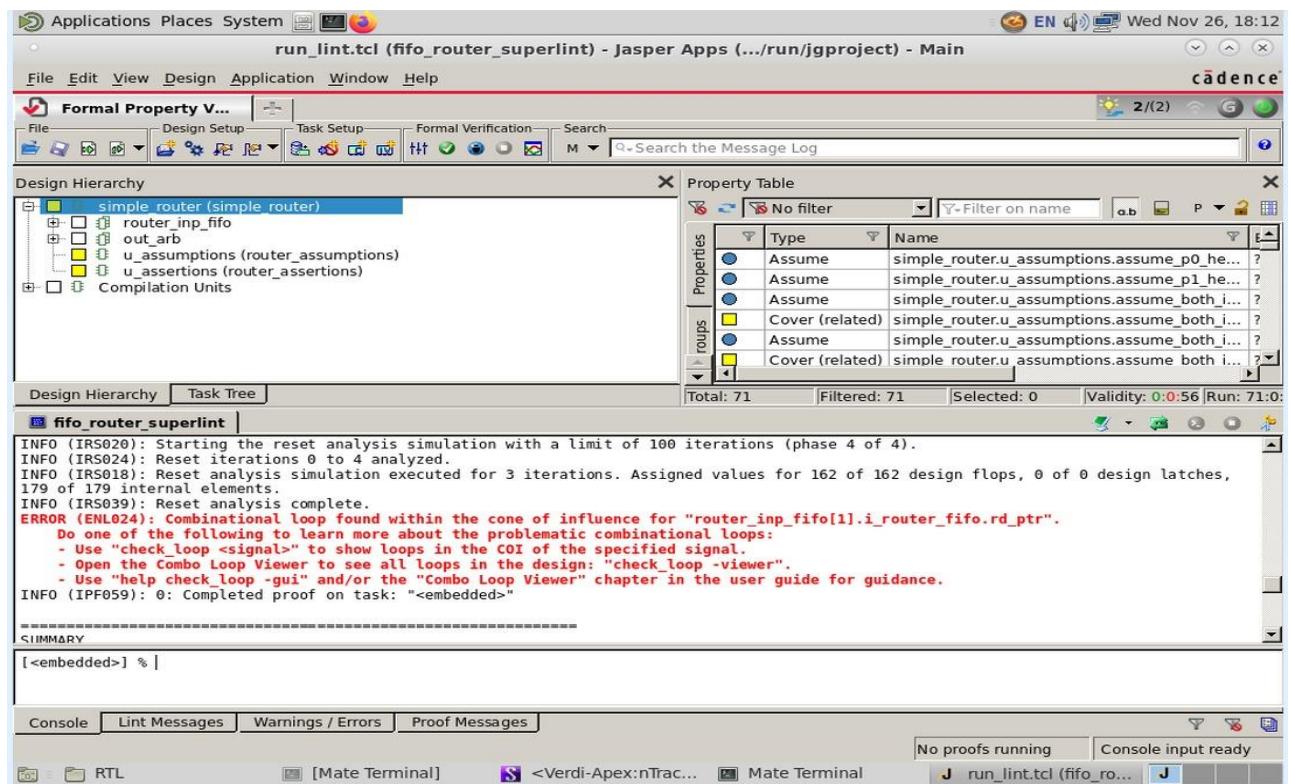
Task	Property Summary	Assertion	Cover	Constraint	Vacuity	Witness
FXP	Total Properties: 2 - passed[2] - failed[0] - disabled[0]; Constraints Enabled: 17; Run Time: 0:00:07	fpx_1_out_fifo_full_0 fpx_1_out_pie_out_1	...pie_router.sv:9 ...pie_router.sv:6	not_computed not_computed	- -	t1 t1
- Bottom Status Bar:** Total Properties: 2 - passed[2] - failed[0] - disabled[0]; Constraints Enabled: 17; Run Time: 0:00:07.
- VC Formal Console:**

```

159 3
160 1
161 1
162 [Info] TW INFO MSG: Starting rewrite (rw1_1) command at 2025-11-25:21:45:21.
163 [Info] BITLEVEL MODEL STATUS: Generated model with 169 gates, 78 inputs, 54 registers, 0 initial constraints, 13 constraints.
164 [Info] BITLEVEL MODEL STATUS: Regs and rw1_1 gates: 169, reg: 54, initialConst: 0, invarConst: 13.
165 [Info] TW INFO MSG: Finished rewrite (rw1_1) command in 1 seconds.
166 [Info] TW INFO MSG: Starting v3 refine (rw1_1) command at 2025-11-25:21:45:23.
167 [Info] TW INFO MSG: Finished refine (rw1_1) command in 1 seconds.
168 #check_fv -stop
    
```
- Bottom Status Bar:** run\_fpvtcl (~/Document...), run, run\_fxp.tcl (~/Document...), Mate Terminal, <Verdi-Apex:nTraceM...>.

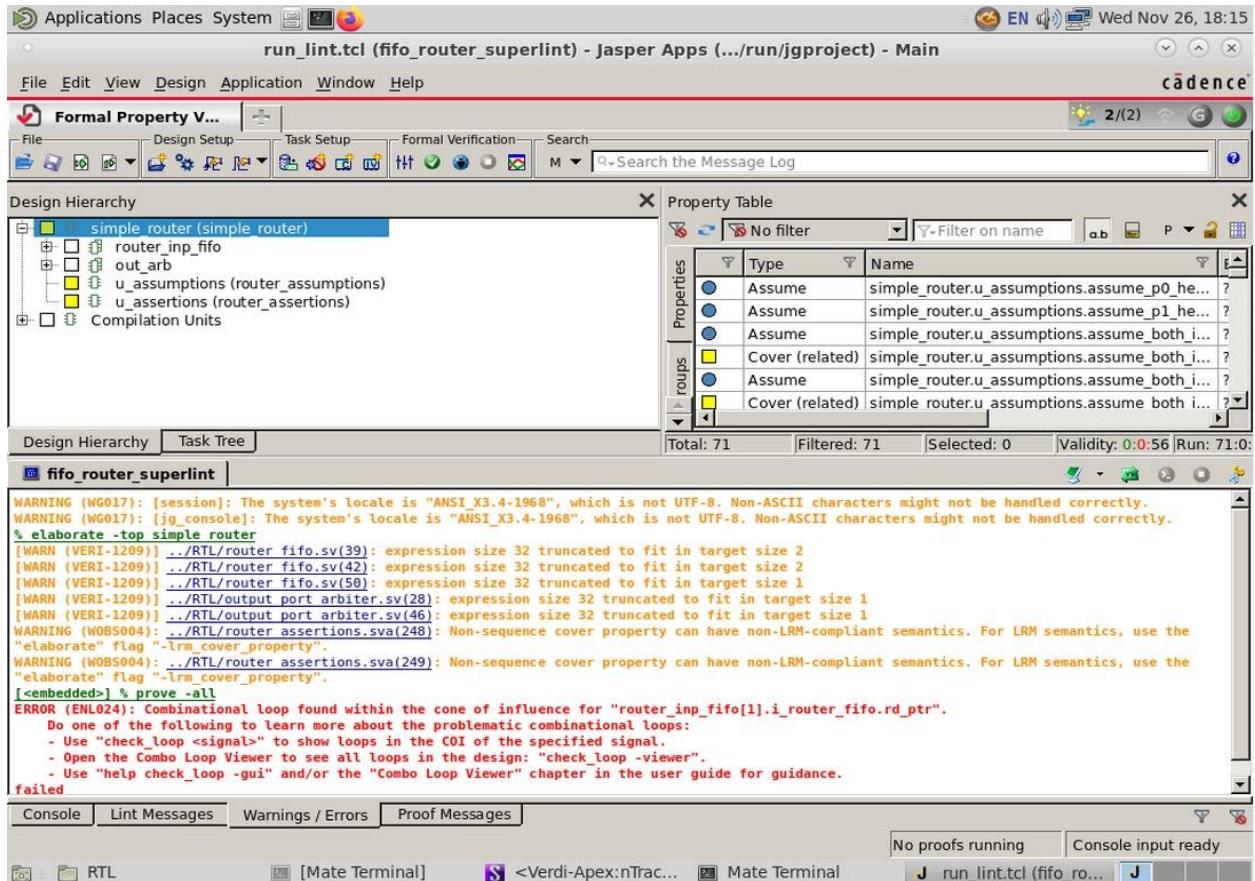
## Part -3 Formal Verification in Jasper Gold.

After running the given design



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

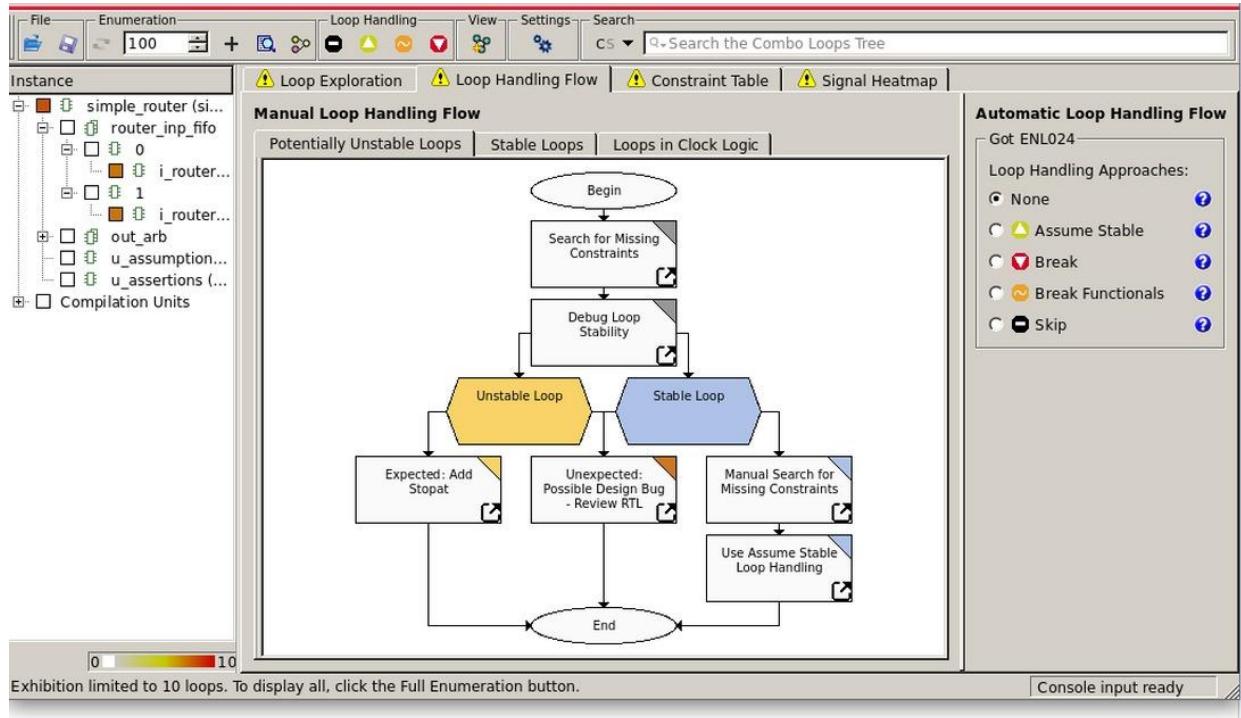
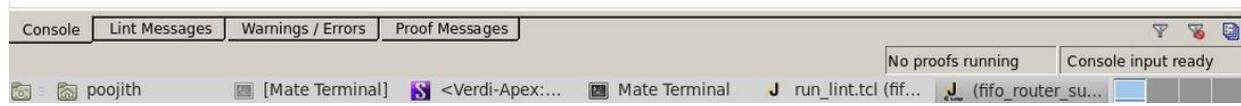
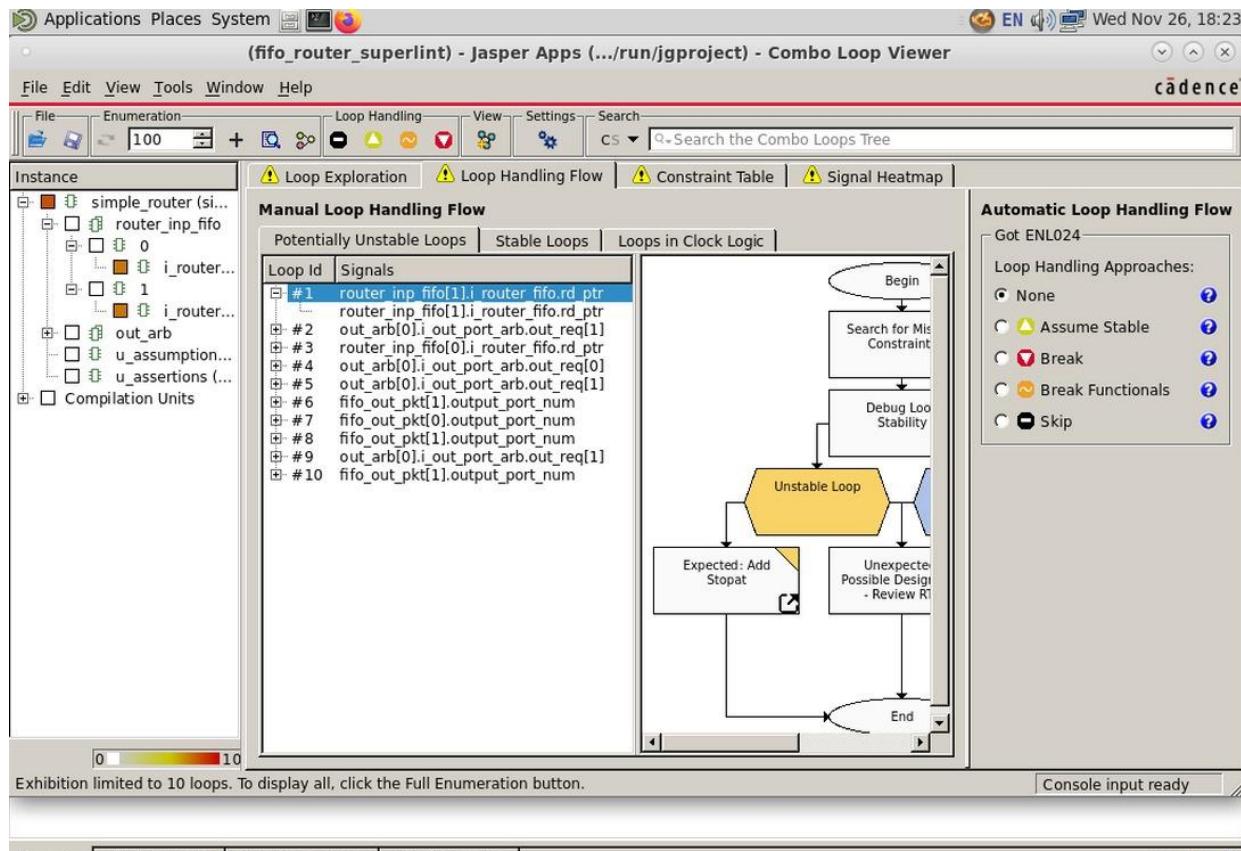
This error was detected in Jasper gold and not in VC Formal, therefore Jasper Gold is slightly better as it detects combinational loops. In VC Formal we did run it through AEP, but we cannot find it in the AEP app.

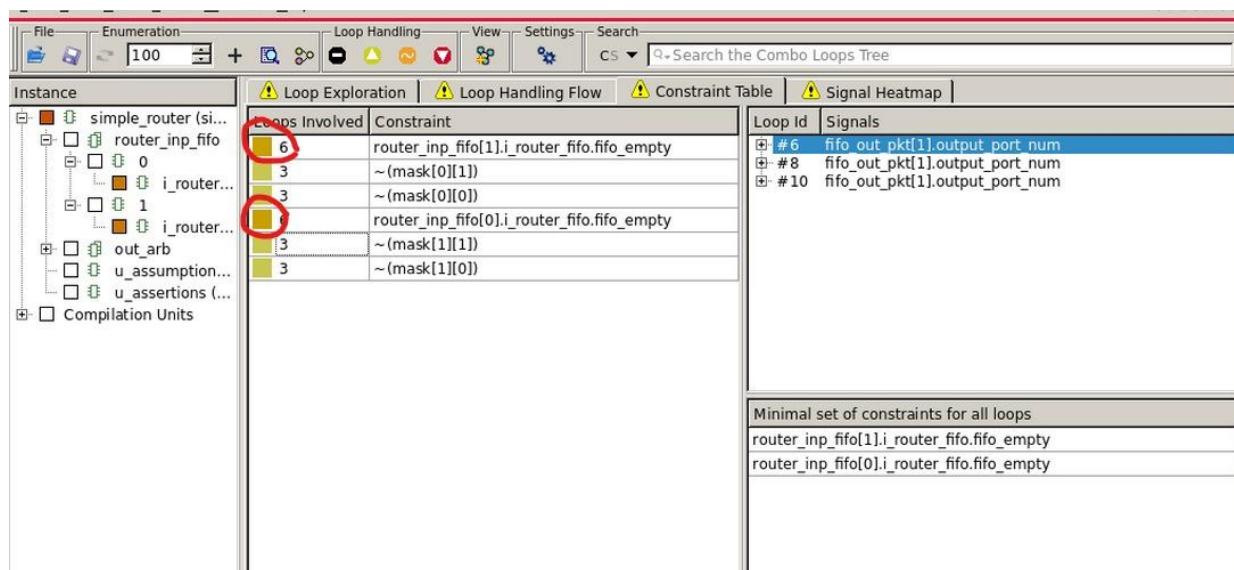
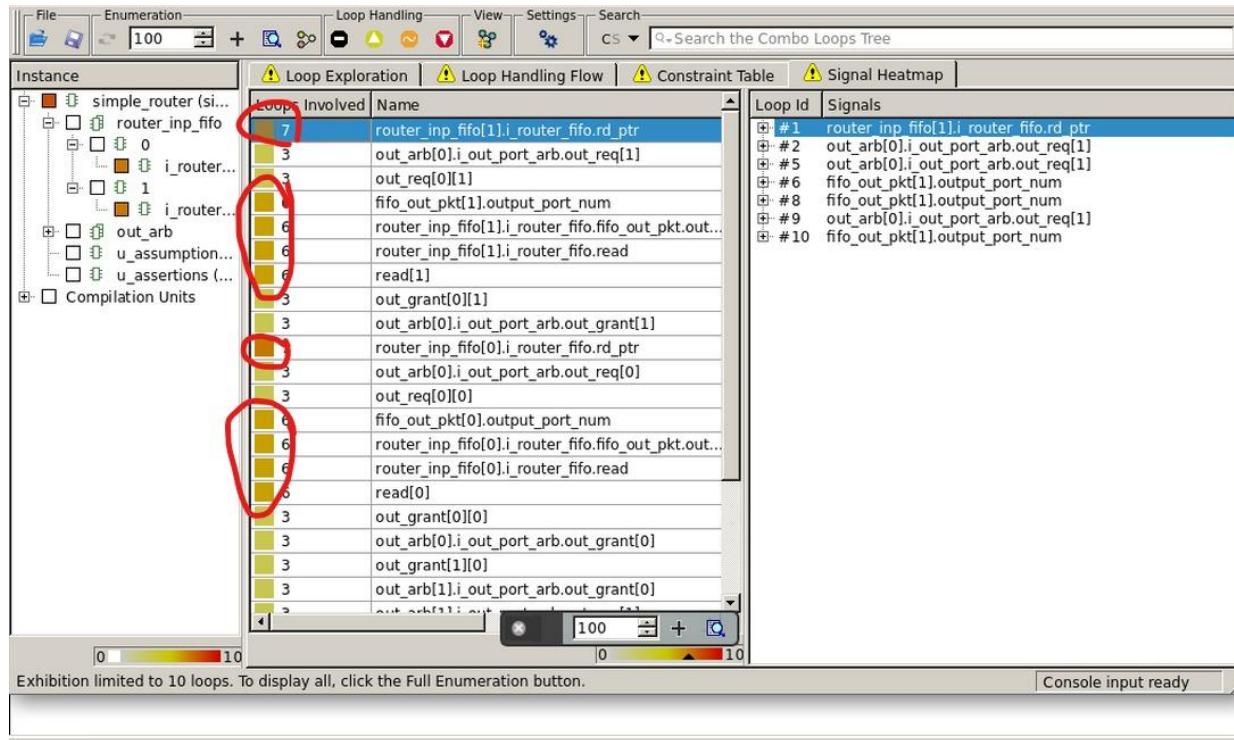


We are assigning a larger width bit to the smaller width target bit in the router fifo.

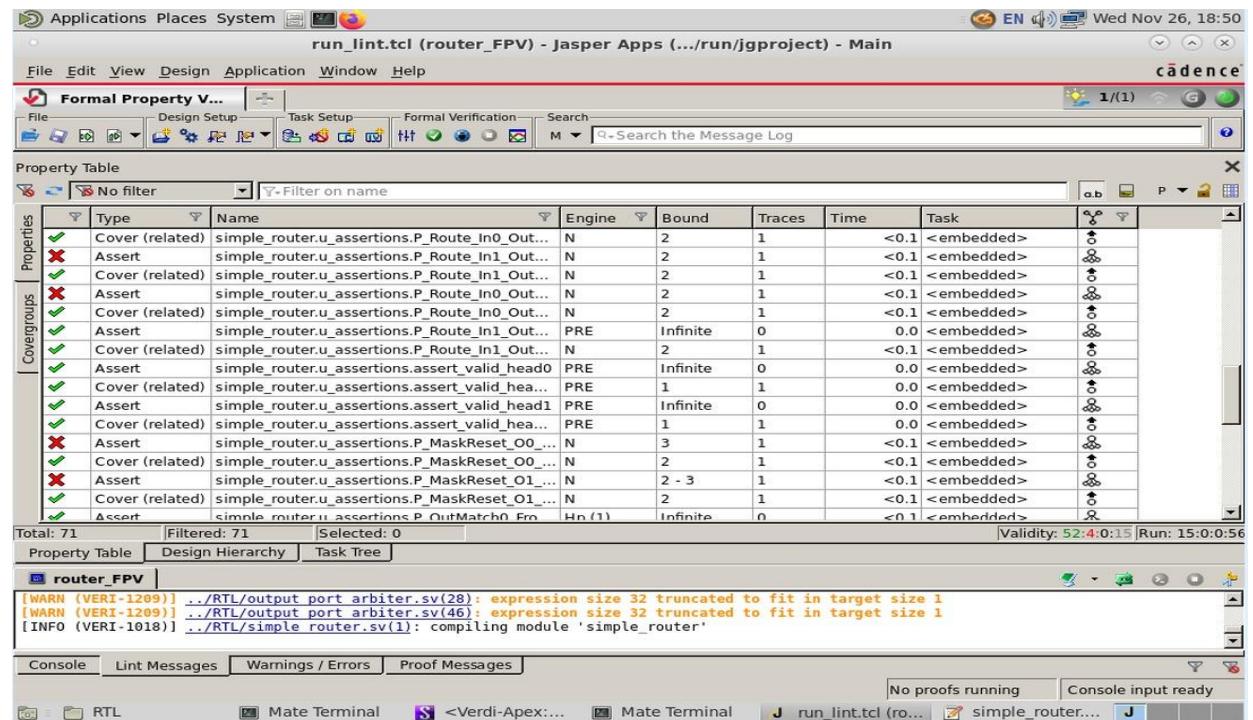
This warning means that some of my cover properties are written in a complex style, so Jasper is interpreting them using its own default behavior instead of the exact official SystemVerilog standard. Jasper suggests using the `-lrm_cover_property` option if strict standard behavior is required.

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.

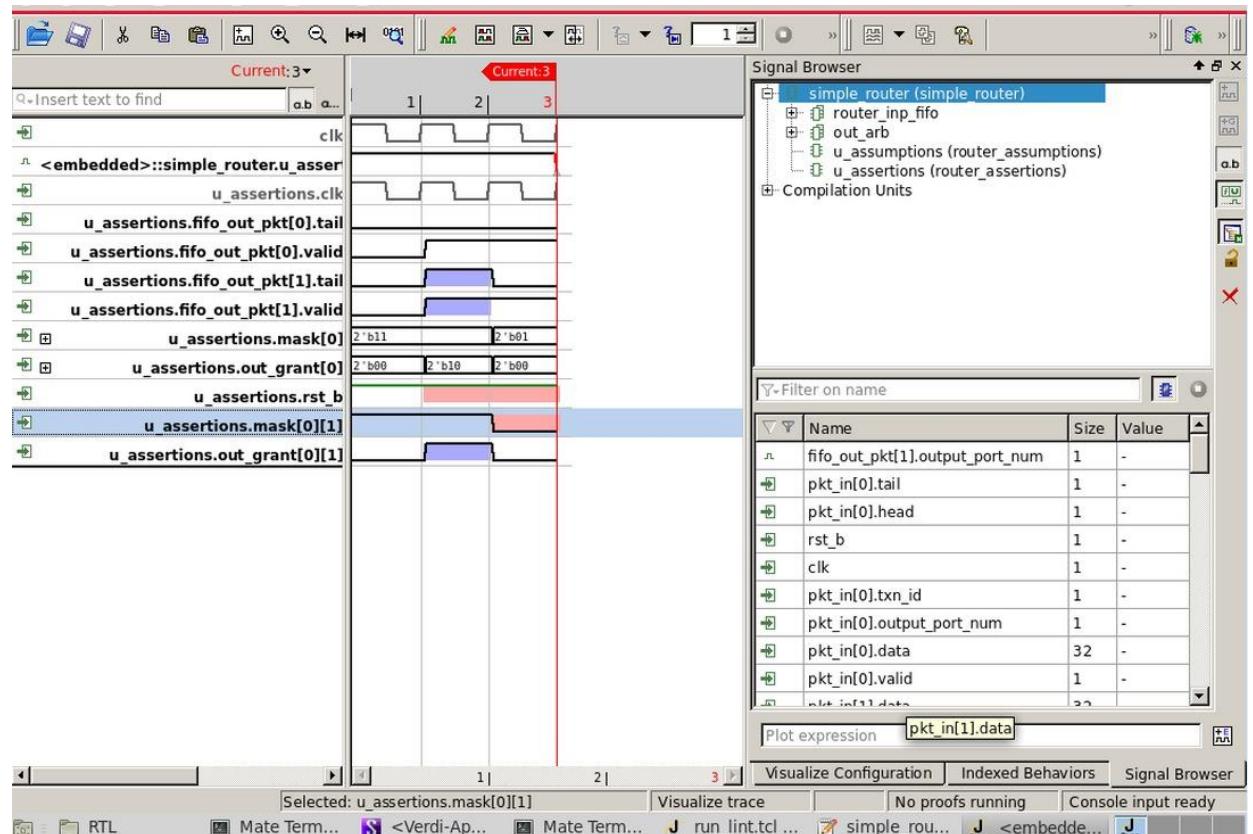


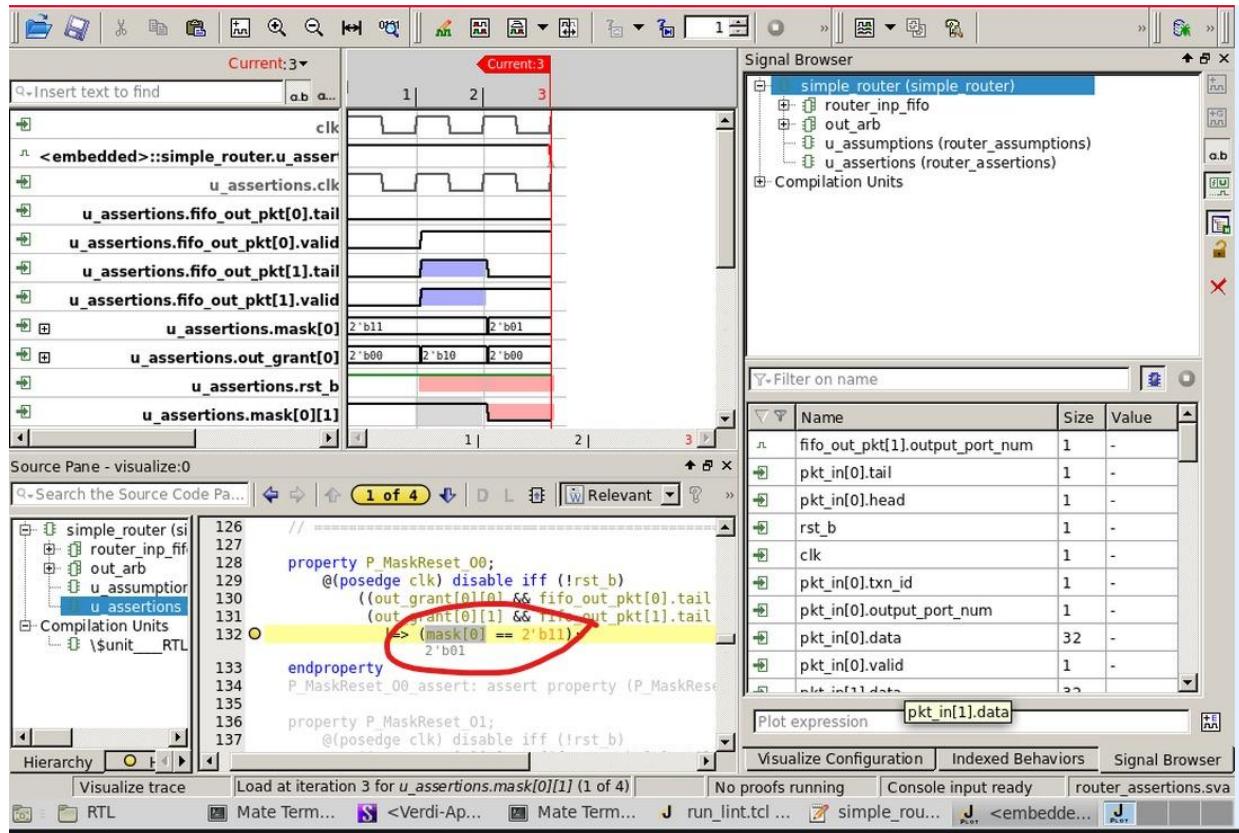


## Bug Identification1:



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:

```

58     mask[1] <= 1;
59   end
60 else begin
61   if((out_grant[0][0] && fifo_out_pkt[0].tail) || (out_grant[0][1] && fifo_out_pkt[1].tail)) begin
62     mask[0] <= 2'h1; //before bug fix
63   end
64 else if (out_grant[0][0]) begin
65   mask[0] <= 2'h1;
66 end
67 else if(out_grant[0][1]) begin
68   mask[0] <= 2'h2;

```

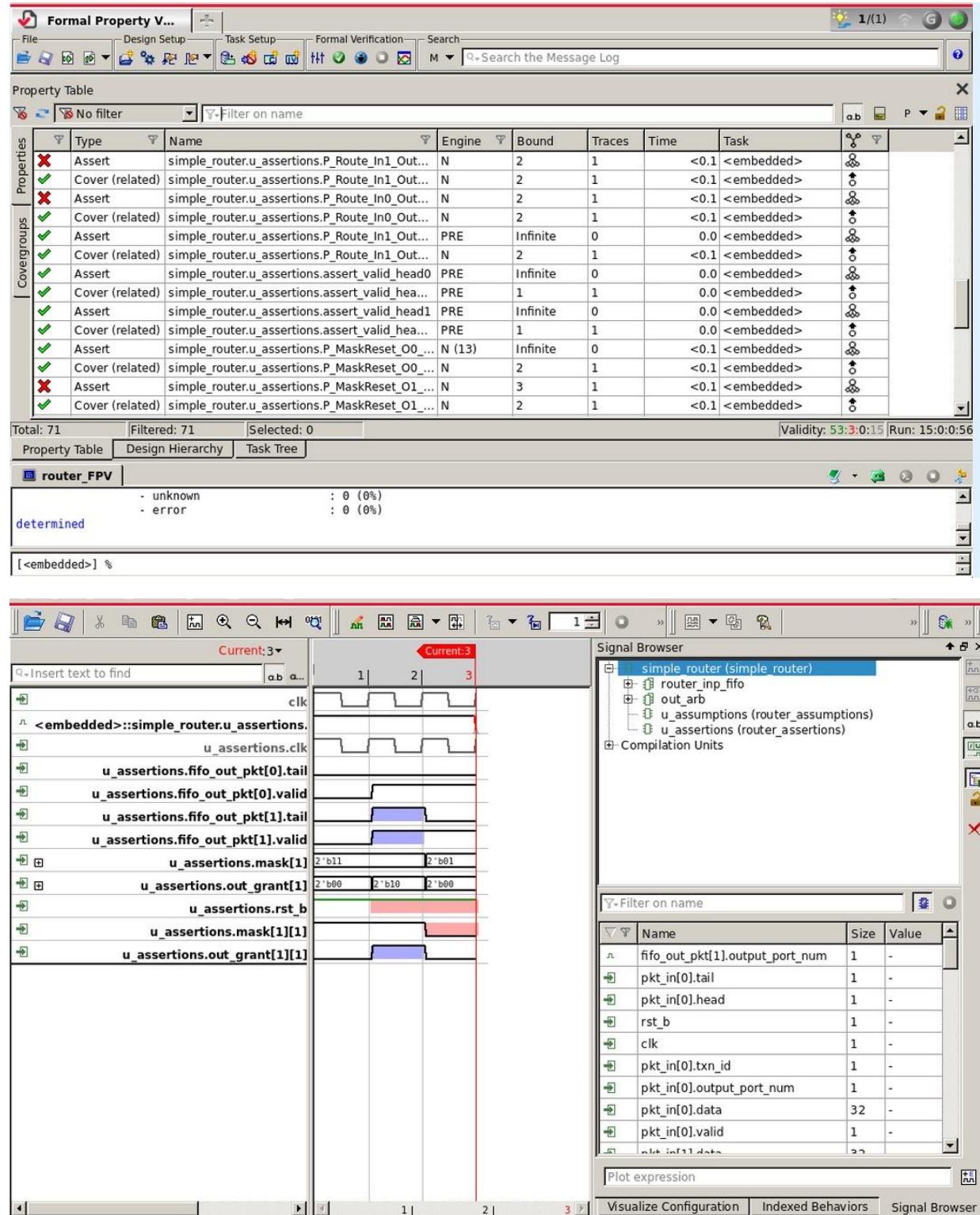
### After bug Fix:

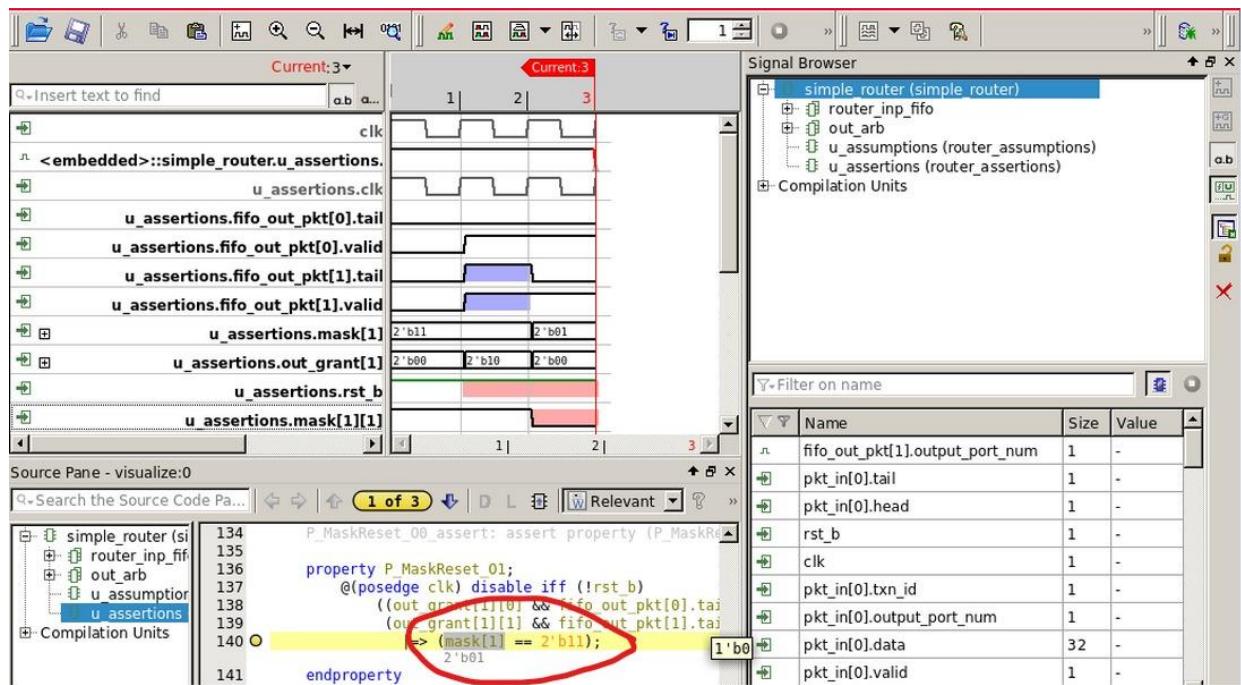
```

58     mask[0] <= 1;
59     mask[1] <= 1;
60   end
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; // bug fix
64   end
65 else if (out_grant[0][0]) begin
66   mask[0] <= 2'h1;
67 end
68 else if(out_grant[0][1]) begin
69   mask[0] <= 2'h2;

```

## 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:

```

68     mask[0] <= 2'h2;
69 end
70
71 if((out_grant[1][0] && fifo_out_pkt[0].tail) || (out_grant[1][1] && fifo_out_pkt[1].tail)) begin
72     mask[1] <= 2'h1;    // bug
73 end
74 else if (out_grant[1][0]) begin
75     mask[1] <= 2'h1;
76 end
77 else if(out_grant[1][1]) begin

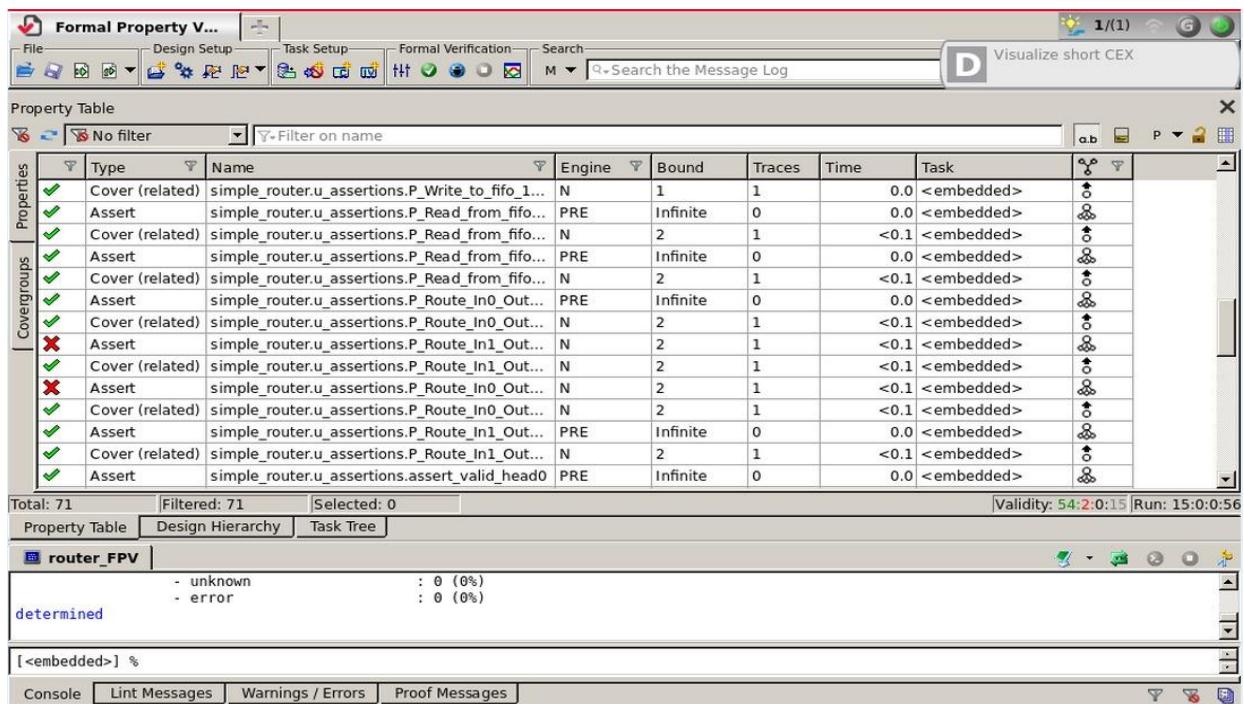
```

### After Bug Fix:

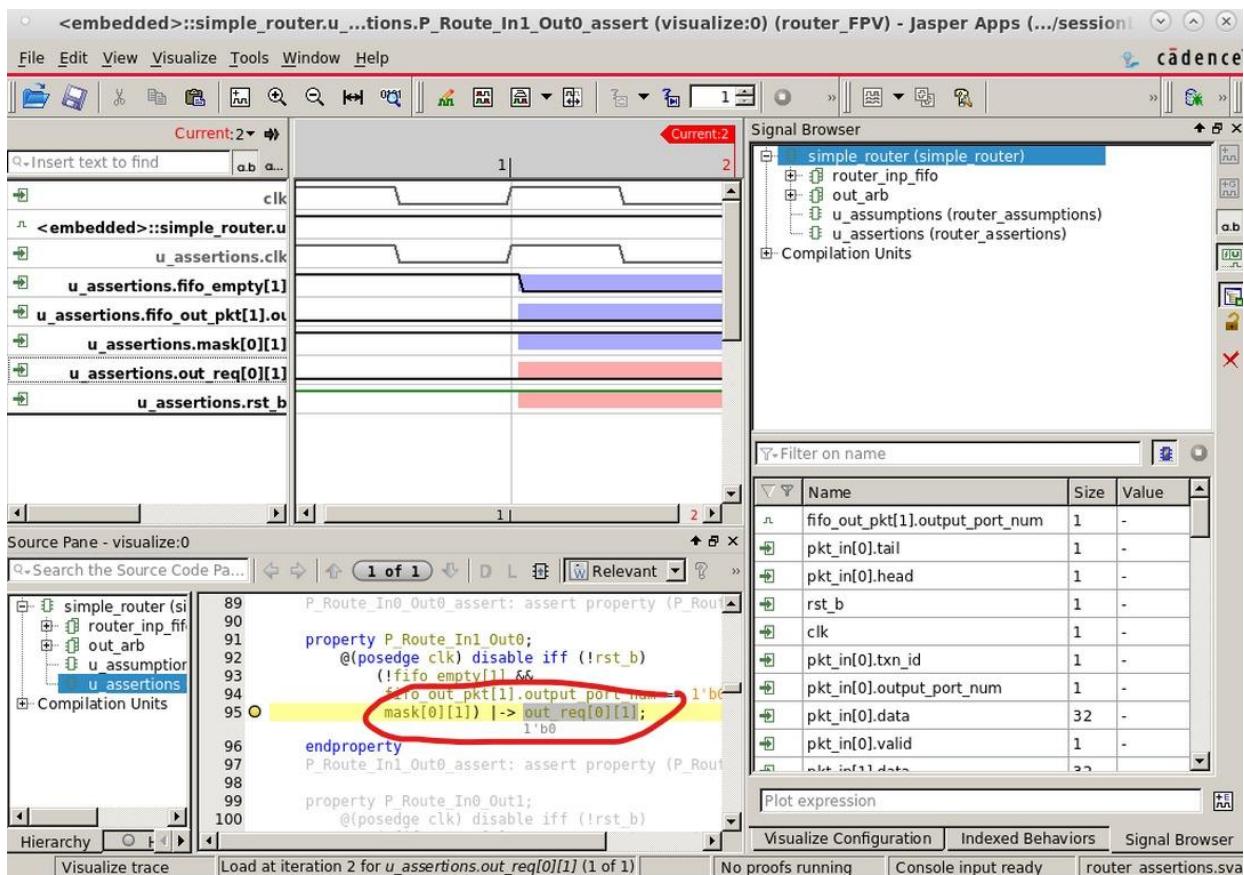
```

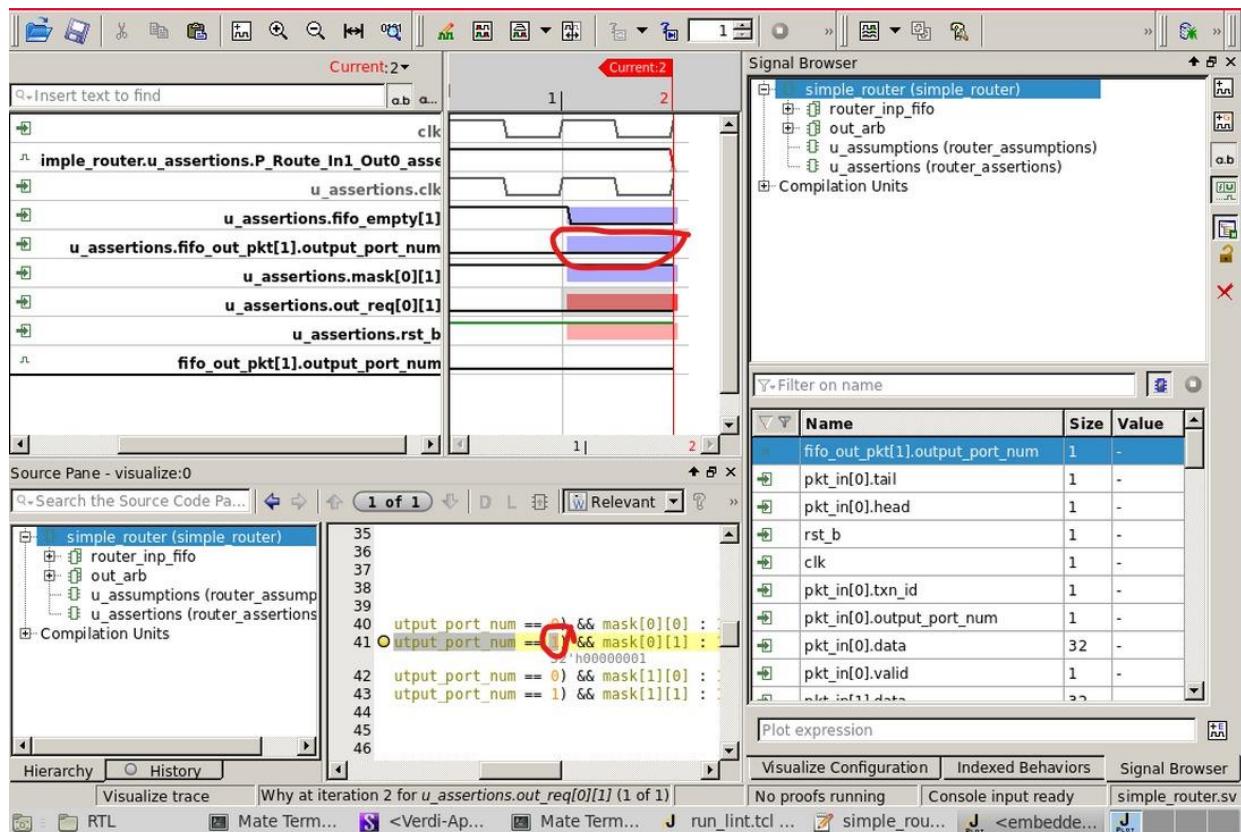
69 end
70
71 if((out_grant[1][0] && fifo_out_pkt[0].tail) || (out_grant[1][1] && fifo_out_pkt[1].tail)) begin
72     mask[1] <= 2'h1;    // bug fix
73 end
74 else if (out_grant[1][0]) begin
75     mask[1] <= 2'h1;
76 end
77 else if(out_grant[1][1]) begin

```



### 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:

```

38 endgenerate
39
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; // bug
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
45

```

### After Bug Fix:

```

37 end : router_inp_if
38 endgenerate
39
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 == 0) && mask[0][1] : 1'b0; // bug fix
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
45

```

## Bug Identification 4:

**Formal Property Verification (FPV) Results:**

Total: 71 | Filtered: 71 | Selected: 0 | Validity: 55:1:0:15 | Run: 15:0:0:56

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_head0	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_head1	PRE	1	1	0.0	<embedded>
Assert	simple_router.u_assertions.P_MaskReset_O0...	N (13)	Infinite	0	<0.1	<embedded>

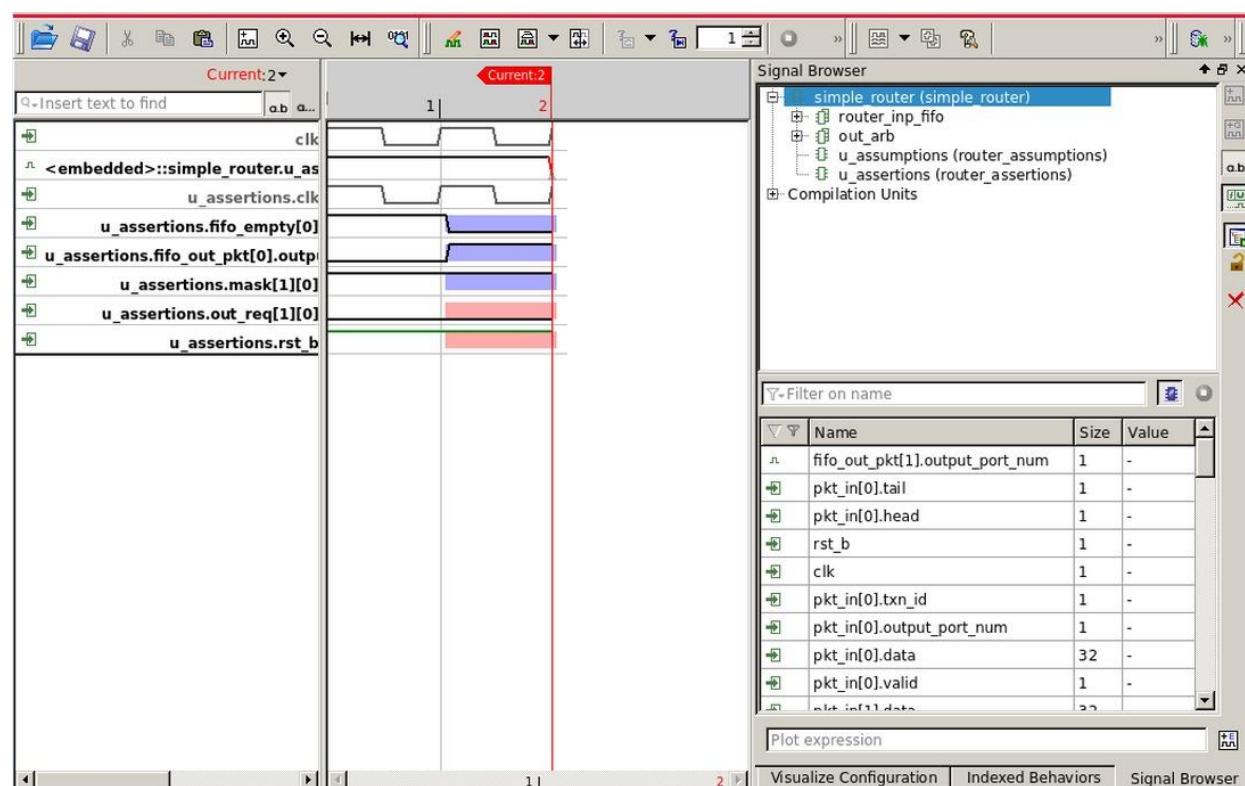
**Router FPV Status:**

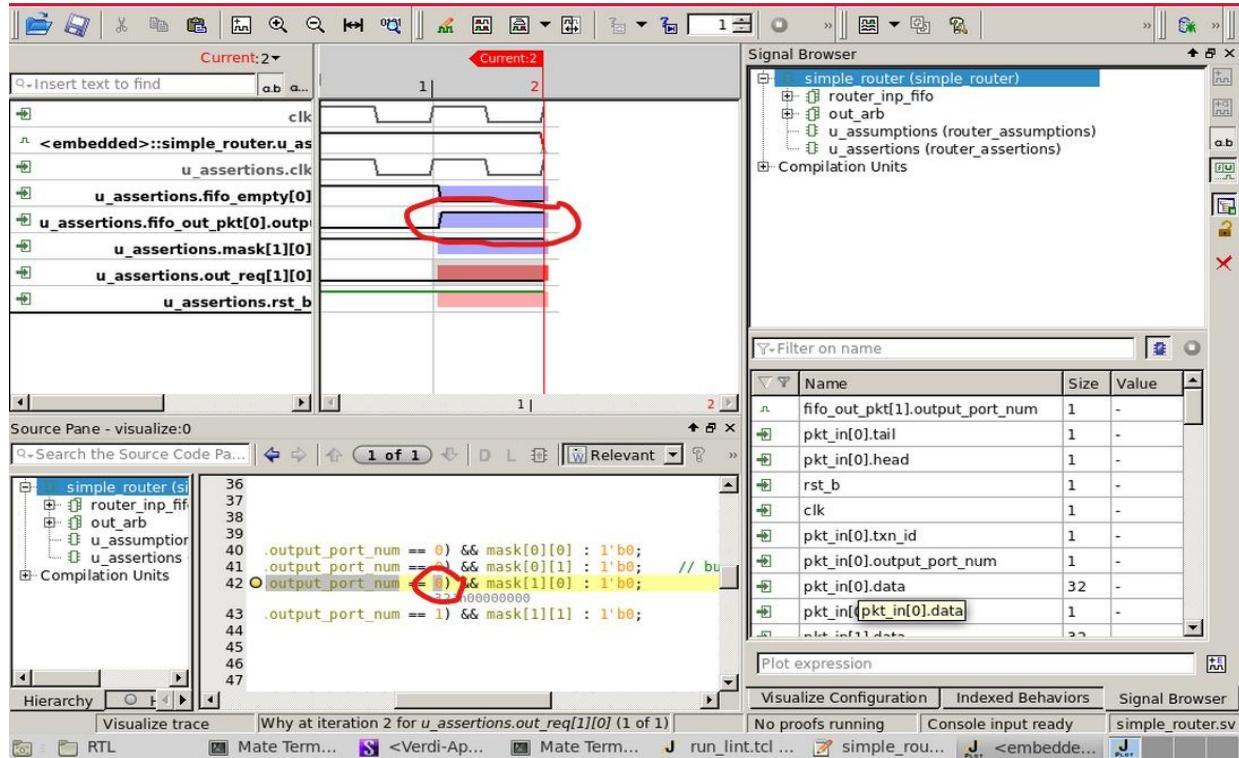
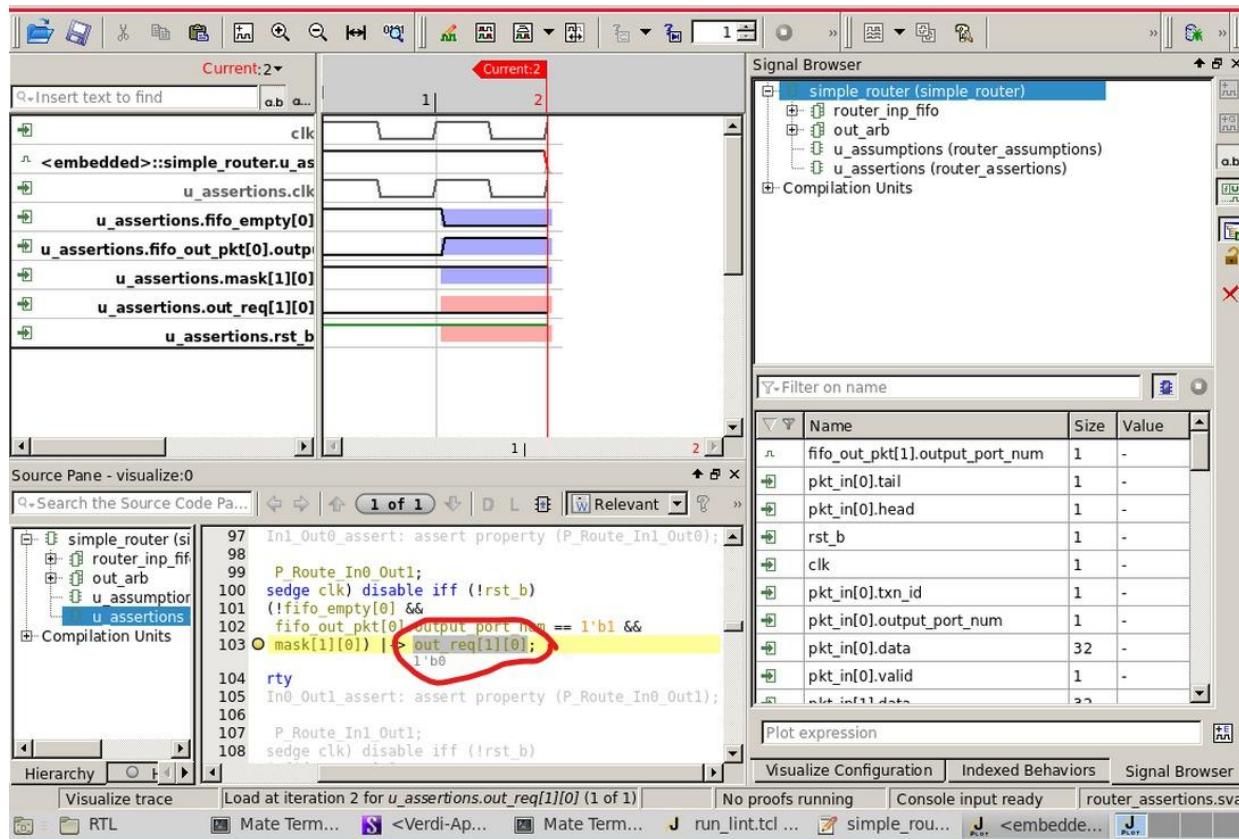
- unknown : 0 (0%)
- error : 0 (0%)

**Determined:** [ ] %

**Console Tab:** Lint Messages | Warnings / Errors | Proof Messages | No proofs running | Console input ready

**Tool Buttons:** RTL, Mate Terminal, Verdi-Apex..., Mate Terminal, run\_lint.tcl (ro...), simple\_router..., Signal Browser





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

### Before Bug Fix:

```
38 endgenerate  
39  
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 == 0) && mask[0][1] : 1'b0; // bug fix  
42 assign out_req[1][0] = !fifo_empty[0] ? (fifo_out_pkt[0].output_port_num == 0) && mask[1][0] : 1'b0; // bug  
43 assign out_req[1][1] = !fifo_empty[1] ? (fifo_out_pkt[1].output_port_num == 1) && mask[1][1] : 1'b0;  
44
```

### After Bug Fix:

```
38 endgenerate  
39  
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 == 0) && mask[0][1] : 1'b0; // bug fix  
42 assign out_req[1][0] = !fifo_empty[0] ? (fifo_out_pkt[0].output_port_num == 0) && mask[1][0] : 1'b0; // bug  
43 assign out_req[1][1] = !fifo_empty[1] ? (fifo_out_pkt[1].output_port_num == 1) && mask[1][1] : 1'b0;  
44
```

### FPV Final Simulation Output:

The screenshot shows the Formal Property Verification (FPV) software interface. The top menu bar includes 'File', 'Design Setup', 'Task Setup', 'Formal Verification', and 'Search'. The main window has three tabs: 'Property Table', 'Design Hierarchy', and 'Task Tree'. The 'Design Hierarchy' tab is active, displaying a tree structure under 'simple\_router (simple\_router)': router\_inp\_fifo, out\_arb, u\_assumptions (router\_assumptions), u\_assertions (router\_assertions), and Compilation Units. The 'Property Table' tab shows a table for 'router\_FPV' with four rows: ar\_covered (0%, 0%), undetermined (0%, 0%), unknown (0%, 0%), and error (0%, 0%). The status bar at the bottom indicates '[<embedded>] %'.

- ar_covered	:	0 (0%)
- undetermined	:	0 (0%)
- unknown	:	0 (0%)
- error	:	0 (0%)

determined

[<embedded>] %

**Formal Property V...**

File Design Setup Task Setup Formal Verification Search M Search the Message Log

Property Table

No filter Filter on name

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>
Cover (related)	simple_router.u_assumptions.assume_mid_fli...	POE	1	1	0.0	<embedded>

Total: 71 Filtered: 71 Selected: 0 Validity: 56:0:0:15 Run: 15:0:0:56

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 M Search the Message Log

Task Tree

Name	Result
All Tasks	 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):

```
1
2 =====
3 Jasper Verification Results
4 =====
5 2024.06p002 64 bits for Linux64 3.10.0-1160.21.1.el7.x86_64
6 Host Name: mo_ece.pdx.edu
7 User Name: poojith
8 Printed on: Wednesday, Nov26, 2025 08:14:30 PM PST
9 Working Directory: /u/poojith/JD_M/HW3/run
0
1
2 =====
3 DETAILED RESULTS
4 =====
5
6 ---[ <embedded> ]-----
7 [1]
8   Name      : simple_router.u_assumptions.assume_p0_head_tail_exclusive
9   Expression : u_assumptions.assume_p0_head_tail_exclusive
10  Type      : assume
11  Location   : ../RTL/router_assumptions.sva, 16
12
13  Status     : temporary
14 [2]
15  Name      : simple_router.u_assumptions.assume_p1_head_tail_exclusive
16  Expression : u_assumptions.assume_p1_head_tail_exclusive
17  Type      : assume
18  Location   : ../RTL/router_assumptions.sva, 23
19
20  Status     : temporary
21 [3]
22  Name      : simple_router.u_assumptions.assume_both_inputs_valid_0
23  Expression : u_assumptions.assume_both_inputs_valid_0
24  Type      : assume
```

X-Propagation Ver... X-Prop

File Design Setup X-Prop Formal Verification Graph Tools Search M 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

X-Propagation Analysis Browser Waiver List

router\_FPV

SUMMARY

Properties Considered

assertions	:	56
- proven	:	22
- 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%)
undetermined	:	0 (0%)

[<embedded>] %

Console Lint Messages Warnings / Errors Proof Messages

No proofs running Console input ready

run Mate Term <Verdi-An Mate Term run\_lint\_tcl simple\_rout Reset Anal

```
1 module simple_router()
2 input logic clk,
3 input logic rst_b,
4
```