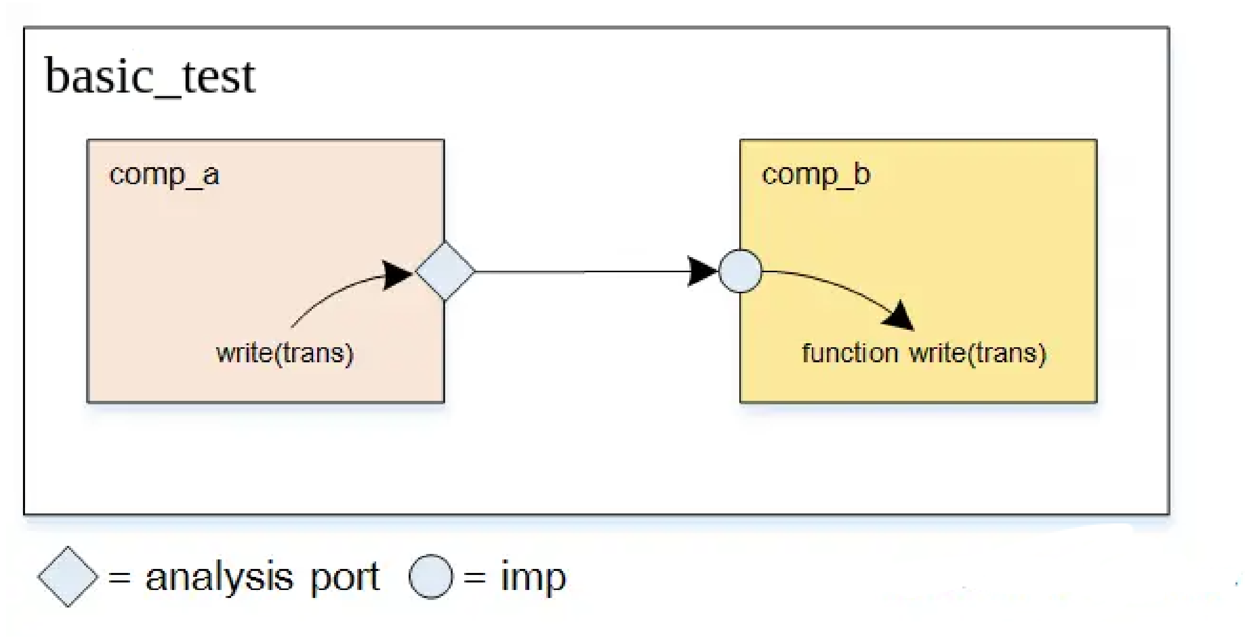
TLM Analysis port Analysis imp port



TLM Analysis port and analysis imp port enable broadcasting a transaction to one or many components. This implementation shows connecting an analysis port to an analysis imp port.

Implementing analysis port in comp_a

```
//Step-1. Declaring analysis port
  uvm_analysis_port#(transaction) analysis_port;

//Step-2. Creating analysis port
    analysis_port = new("analysis_port", this);

//Ste-3. Calling write method
    analysis_port.write(trans);
```

Verify the above steps in component_a class and also observe the places where these steps have been implemented.

Implementing analysis imp_port in comp_b

```
//Step-1. Declaring analysis imp port
  uvm_analysis_imp#(transaction,component_b) analysis_imp;

//Step-2. Creating analysis imp_port
    analysis_imp = new("analysis_imp", this);

//Step-3. Implementing write method
  virtual function void write(transaction trans);
    `uvm_info(get_type_name(),$sformatf(" Inside write method.
Received trans On Analysis Imp Port"),UVM_LOW)
    `uvm_info(get_type_name(),$sformatf(" Printing trans, \n
%s",trans.sprint()),UVM_LOW)
  endfunction
```

Verify the above steps in component_b class and also observe the places where these steps have been implemented.

Connecting analysis port and analysis imp_port in basic_test

```
function void connect_phase(uvm_phase phase);
  //Connecting analysis port to imp port
  comp_a.analysis_port.connect(comp_b.analysis_imp);
endfunction : connect_phase
```

Run the *basic_test.sv* file and observe the output. Transaction should be sent from component_a to component_b.

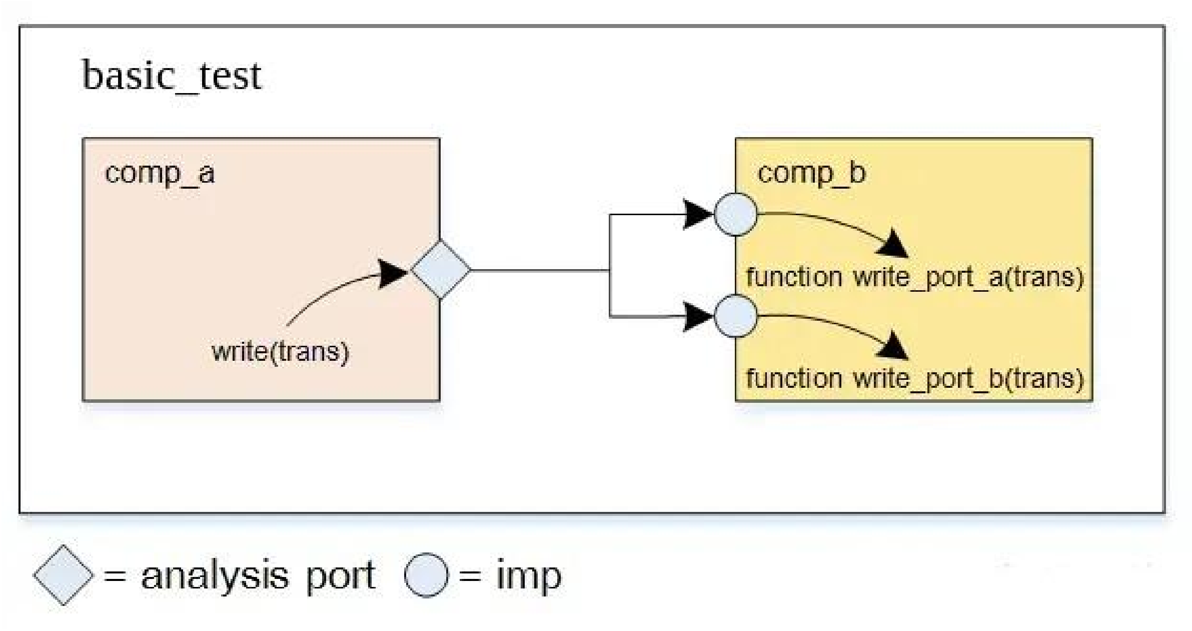
Task 1

Submit the screenshot of output of the above run.

For next tasks Submission
Make a directory with the name “Task2/Task3/Task4” and move all the code files in them and then start working (some files will remain the same). (The whole folder along with the snapshot of the output are the submission for the respective task)

Task 2

TLM Analysis port multi Analysis imp port



This task requires connecting analysis port to multiple analysis imp ports.

comp__a
This component will remain the same for this task.

Implementing analysis imp__ports in comp__b

This component will be implemented by following the below mentioned steps

1. define the analysis__imp__ports using macro

```
`uvm_analysis_imp_decl(port_identifier)

//port_identifier is the user defined name to differentiate the
imp ports
```

defining ports with the name __port__a and __port__b

```
`uvm_analysis_imp_decl(_port_a)

`uvm_analysis_imp_decl(_port_b)
```

2. Declare the analysis ports

```
uvm_analysis_imp<> #(t,T) port_name;
```

declaring two ports using the above syntax,

```
uvm_analysis_imp_port_a #(transaction,component_b)
analysis_imp_a;

uvm_analysis_imp_port_b #(transaction,component_b)
analysis_imp_b;
```

3. Creating the analysis ports

```
    analysis_imp_a = new("analysis_imp_a", this);

    analysis_imp_b = new("analysis_imp_b", this);
```

4. Implement the write methods for each port,

```
function void write_port_identifier;
```

Implementing write methods with the above syntax,

Method for analysis_imp_a

```
virtual function void write_port_a(transaction trans);
  `uvm_info(get_type_name(),$sformatf(" Inside write_port_a
method.Received trans On Analysis Imp Port"),UVM_LOW)
  `uvm_info(get_type_name(),$sformatf(" Printing trans, \n
%s",trans.sprint()),UVM_LOW)
endfunction
```

Method for analysis_imp_b

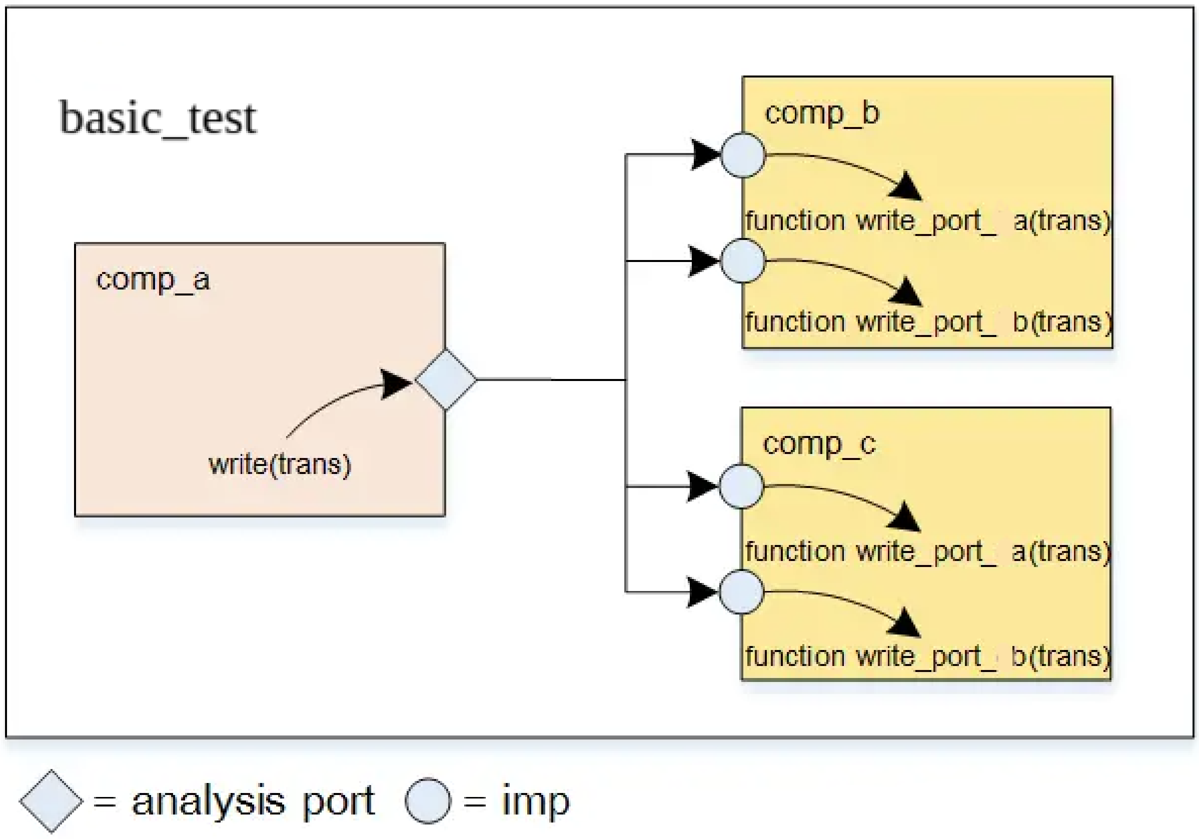
```
virtual function void write_port_b(transaction trans);

  `uvm_info(get_type_name(),$sformatf(" Inside write_port_b
method.Received trans On Analysis Imp Port"),UVM_LOW)

  `uvm_info(get_type_name(),$sformatf(" Printing trans, \n
%s",trans.sprint()),UVM_LOW)

endfunction
```

Connecting Analysis port with the imp_ports in basic_test

```
function void connect_phase(uvm_phase phase);
  //Connecting analysis_port to imp_ports
  comp_a.analysis_port.connect(comp_b.analysis_imp_a);
  comp_a.analysis_port.connect(comp_b.analysis_imp_b);
endfunction : connect_phase
```

Task 3

TLM Analysis port multi Analysis imp port multi component


basic_test
comp_a
write(trans)
comp_b
function write_port_ a(trans)
function write_port_ b(trans)
comp_c
function write_port_ a(trans)
function write_port_ b(trans)
= analysis port
= imp

This task requires connecting the same analysis port to analysis imp ports of multiple components.

comp_a
This component will remain the same for this task.

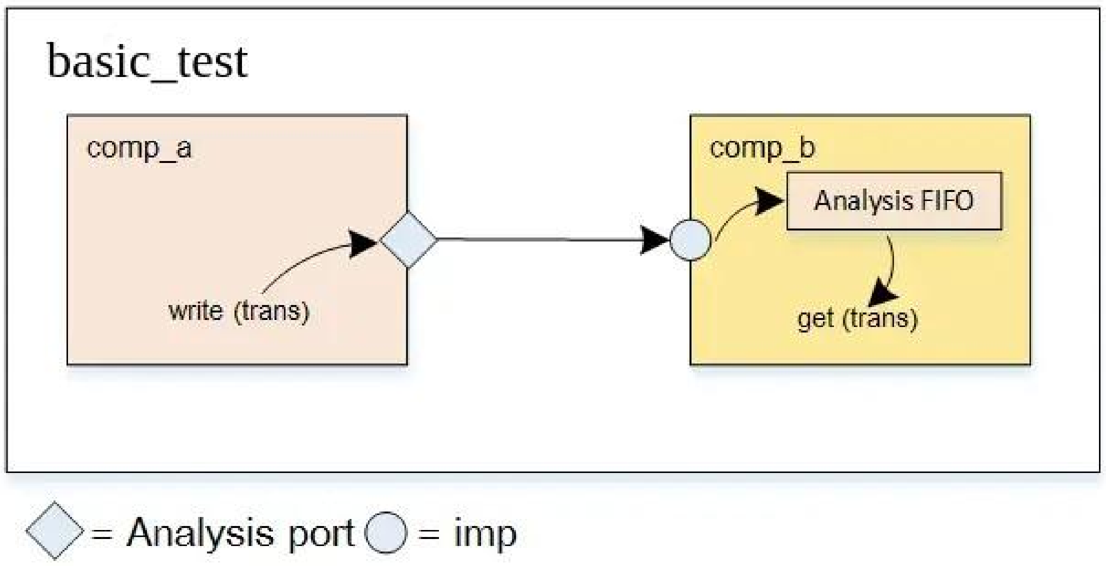
comp_b
This component will remain the same for this task.

Basic_test

For this class, we have to create two objects of comp_b and then connect all the imp ports (total 4, 2 for each object) with the single analysis port of comp_a.

Task 4

TLM Analysis FIFO



TLM Analysis FIFO enables the implementation of FIFO in consumers and connects it directly to the analysis port.

comp_a
This component will remain the same for this task.

comp_b
This component will be implemented by following the below mentioned steps

```
//Step-1. Declaring analysis FIFO
uvm_tlm_analysis_fifo #(transaction) analysis_fifo;
```

```
  //Step-2. Creating analysis FIFO
    analysis_fifo = new("analysis_fifo", this);

  //Step.3 - Getting trans from FIFO (in run_phase)
    analysis_fifo.get(trans);
```

Connecting analysis port and analysis FIFO in basic_test

```
  //Connecting analysis port to analysis FIFO (in connect_phase)

comp_a.analysis_port.connect(comp_b.analysis_fifo.analysis_expor
t);
```