Verification Guide

SystemVerilog Shallow Copy

Shallow Copy

Table of Contents

An object will be created only after doing new to a class handle,

```
packet   pkt_1;
pkt_1  = new();
```

```
packet   pkt_2;
pkt_2  = new pkt_1;
```

In the last statement, pkt_2 is created and class properties were copied from pkt_1 to pkt_2, this is called as “shallow copy”.

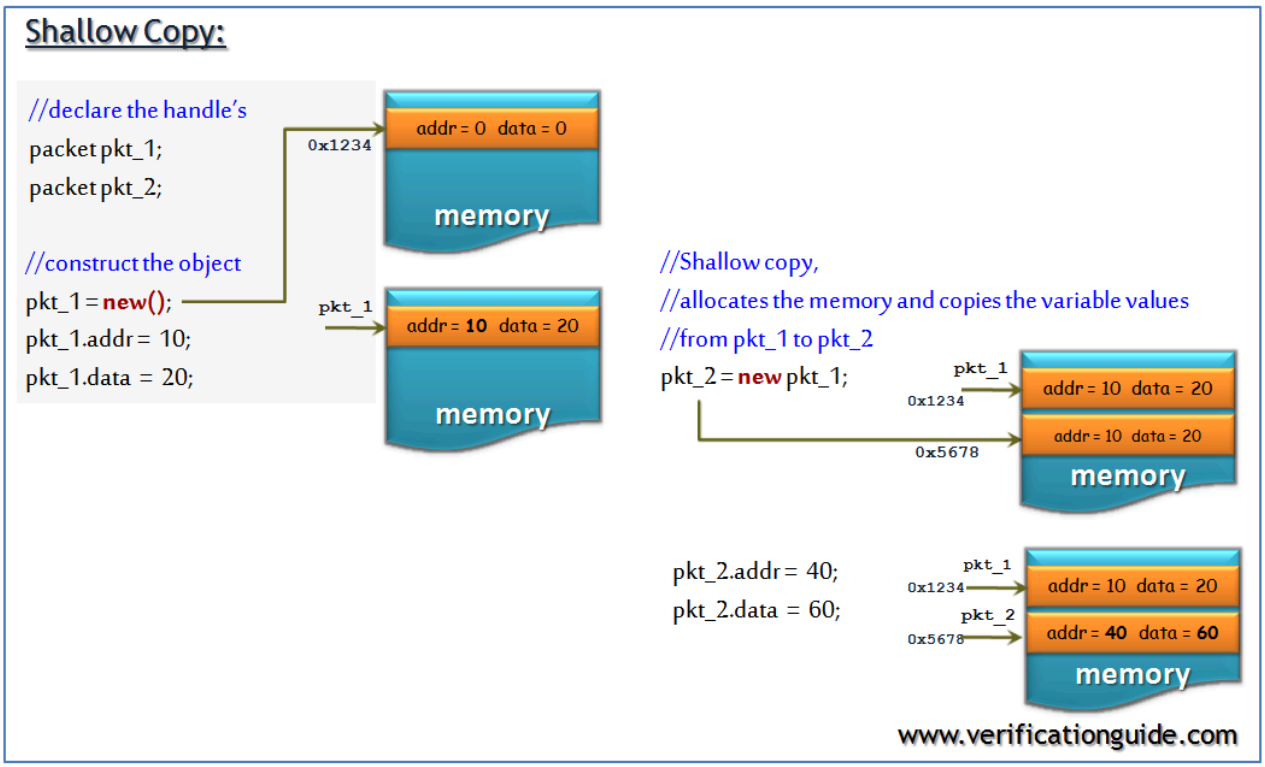
Shallow copy allocates the memory, copies the variable values and returns the memory handle.

In shallow copy, All of the variables are copied across: integers, strings, instance handles, etc.

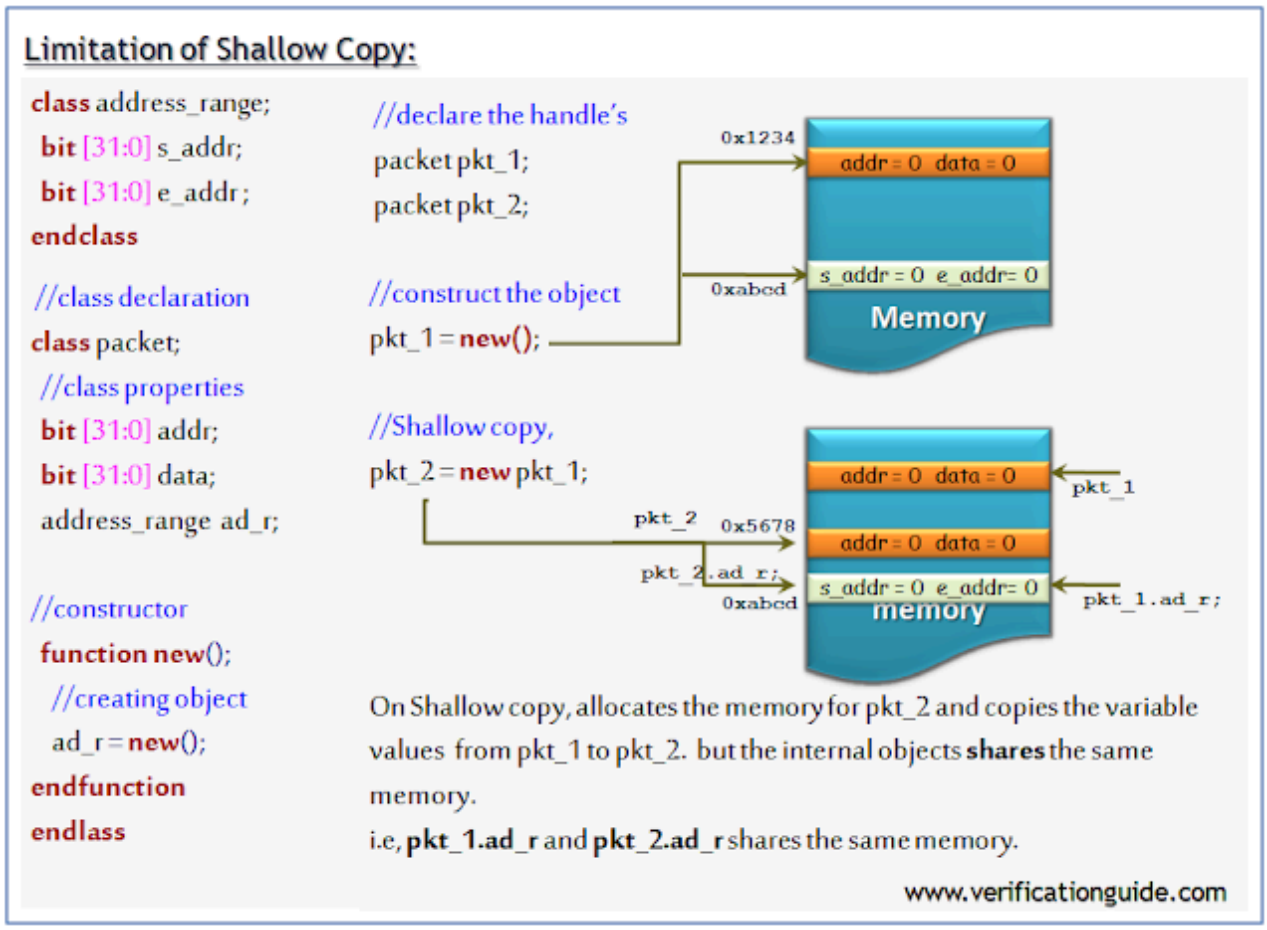
Note::

Objects will not be copied, only their handles will be copied.

to perform the full or deep copy, the custom method can be added.



SystemVerilog Class Shallow Copy



SystemVerilog Shallow Copy Limitation

Shallow copy example

In the below example,

packet class has the properties of bit type and object type (address_range). after the shallow copy addr, data and handle to ar were copied. As it is shallow copy any changes on pkt_2. ar will reflect in pkt_1.ar (because pkt_2.ar and pkt_1.ar will point to the same object).

```
//-- class ---
class address_range;
  bit [31:0] start_address;
  bit [31:0] end_address  ;
  function new();    start_address = 10;
    end_address   = 50;
  endfunction
endclass

//-- class ---
```

```
class packet;
  //class properties
  bit [31:0] addr;
  bit [31:0] data;
  address_range ar; //class handle

  //constructor
  function new();
    addr  = 32'h10;
    data  = 32'hFF;
    ar = new(); //creating object
  endfunction
  //method to display class properties
  function void display();
    $display("------------------------------------------------------------
");
    $display("\t addr  = %0h",addr);
    $display("\t data  = %0h",data);
    $display("\t start_address  = %0d",ar.start_address);
    $display("\t end_address  = %0d",ar.end_address);
    $display("------------------------------------------------------------
");
  endfunction
endclass

// -- module ---
module class_assignment;
  packet pkt_1;
  packet pkt_2;

  initial begin
    pkt_1 = new();   //creating pkt_1 object
    $display("\t****  calling pkt_1 display  ****");
    pkt_1.display();

    pkt_2 = new pkt_1;   //creating pkt_2 object and copying pkt_1 to
pkt_2
    $display("\t****  calling pkt_2 display  ****");
    pkt_2.display();

    //changing values with pkt_2 handle
    pkt_2.addr = 32'h68;
    pkt_2.ar.start_address = 60;
    pkt_2.ar.end_address = 80;
    $display("\t****  calling pkt_1 display after changing pkt_2
properties ****");

    //changes made to pkt_2.ar properties reflected on pkt_1.ar, so only
handle of the object get copied, this is called shallow copy
```

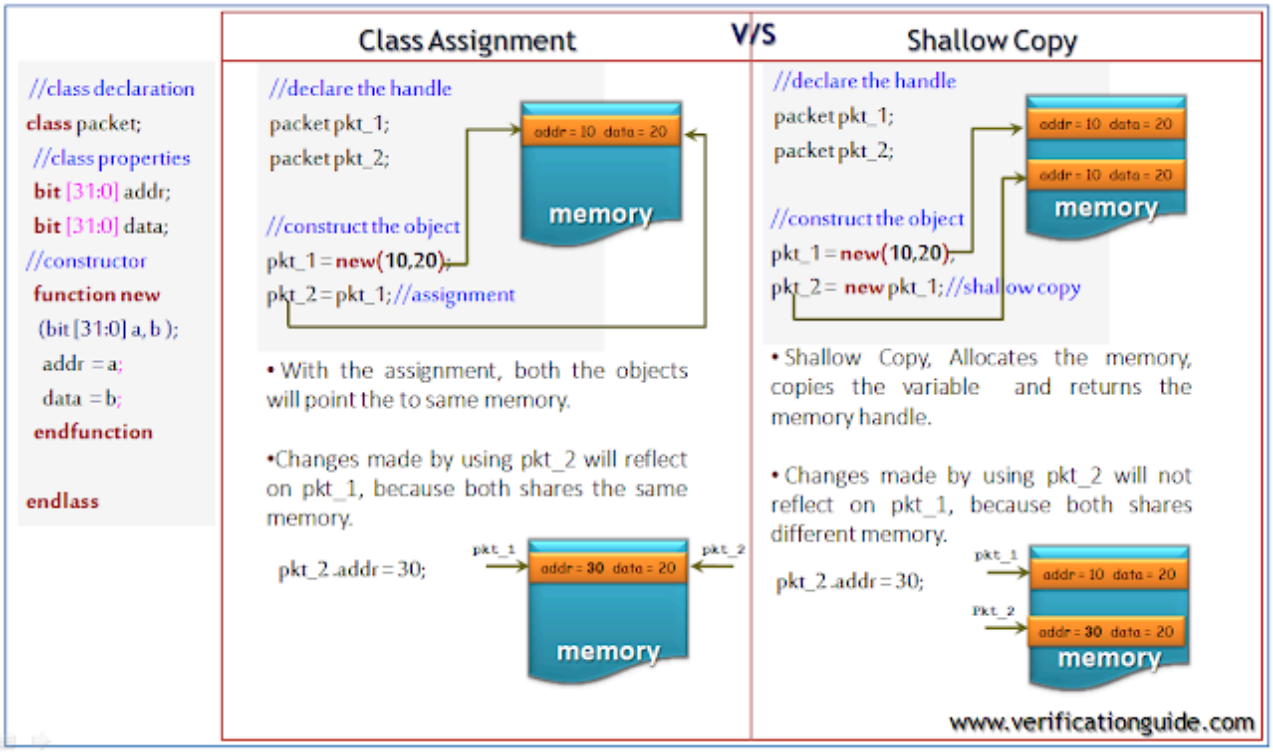
```
    pkt_1.display();
    $display("\t****  calling pkt_2 display after changing pkt_2
properties ****");
    pkt_2.display(); //
  end
endmodule
```

Simulator Output

```
**** calling pkt_1 display ****
-------------------------------------------------------
addr = 10
data = ff
start_address = 10
end_address = 50
-------------------------------------------------------
**** calling pkt_2 display ****
-------------------------------------------------------
addr = 10
data = ff
start_address = 10
end_address = 50
-------------------------------------------------------
**** calling pkt_1 display after changing pkt_2 properties ****
-------------------------------------------------------
addr = 10
data = ff
start_address = 60
end_address = 80
-------------------------------------------------------
**** calling pkt_2 display after changing pkt_2 properties ****
-------------------------------------------------------
addr = 68
data = ff
start_address = 60
end_address = 80
-------------------------------------------------------
```

Click to execute on EDA playground

Difference between Assignment and Shallow Copy



assignment vs shallow copy

‹ Previous

Next ›