LINTING TOOL RULES

DollySoft

LOOKING AHEAD IN TECHLOGY…

Dolly Software Pvt. Ltd.
404, Rishi Bankim Sarani
Hariharpur, Hridaypur,
Barasat, Kolkata-700127
www.dollysoft.co.in.

WELCOME TO THE LINTING TOOL RULES

Rule No 1001 Signal Name Case

Configurable Parameter

Argument type: Signal Name;

Description

Check to see if the signal names, including wires and regs, are all in lower (or upper) case.

```
//////////////// Example: Document.v //////////////////////////

module test (A, b, C);
  input A, b; //warning on 'A' if CASE_LOWER
  output C;
  wire C; //warning on 'C' if CASE_LOWER
  and and1(C,A,b);
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1001: Signals "test.A" cannot be in UPPER CASE.
Violated 1001: Signals "test.C" cannot be in UPPER CASE.
```

Rule No 1002 Signal Name Too Long

Configurable Parameter

Argument type: Signal Length;

Description

Check to see if the length of the signal name exceeds 'length' characters.

```
/////////////// Example : Document.v ////////////////////////

module test (a1234567890123456, b, C);
  input a1234567890123456, b;
  output C;
  wire C;
  and and1(C,a1234567890123456,b);
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1002: Signals "test.a1234567890123456" cannot be greater than "10" characters.
```

Rule No 1003 Clock Name Prefix Suffix

Configurable Parameter

Argument type:Clock Name;

Description

Check Clock name prefix suffix.

```
//////////////// Example : Document.v /////////////////////////

module test (q, clk, en, reset, d);
  output q;
  input clk, en, reset, d;
  reg q;
  wire clk, en, reset, d;
  wire rst_en;
  and U_and_1(rst_en, reset, en);
  always @( posedge clk or negedge rst_en )
    if (rst_en )
      q <= 1'b0;
    else
      q <= d;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1003: Clock signals "test.clk" doesn't follow name conventinon "clk_[_a-zA-Z0-9]*".
```

Rule No 1004 Signal Driven By Multiple devices

Configurable Parameter

Argument type: none;

Description

Cech whether Signal Driven By Multiple devices or not.

```
/////////////// Example : Document.v ////////////////////////

module test (a, b, c, d, f);
  input a, b, c, d;
  output f;
  wire f1, f2;
  assign d = f;
  and and1(f1, a, b);
  or or1(f2, c, d);
  or or2(f, f1, f2);
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1004: Signals test.d is driven by 2 devices.
Violated 1004: Signals test.f is driven by 2 devices.
```

Rule No 1005 Bit Width Mismatch in Assignment

Configurable Parameter

Argument type: none;

Description

The rule checks whether there is any bit width mismatch in the assignment statements.

```
//////////////// Example : Document.v //////////////////////////

module test;
  wire [3:0] a;
  wire [2:0] b;
  assign a = b; //warning on "a" and "b"
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1005: Bit mismatched Signal is "test.b".
```

Rule No 1006 Expression Connected to an Instance Port

Configurable Parameter

Argument type: none;

Description

The rule checks whether there are any logic expressions used in any port connections.

```
/////////////// Example : Document.v ////////////////////////

module test;
  wire a,b1,b2;
  wire c;
  and and1(c, a, b1+b2); //warning on "b1+b2" as port instance;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1006: Some arithmatic operation on and Acctual Gate "and1<0>" which forms another Gate
"_s1".
```

Rule No 1007 Tristate Buffer is not properly defined

Configurable Parameter

Argument type: none;

Description

check whether Tristate Buffer is properly defined or not.

```
/////////////// Example : Document.v ////////////////////////

module Test (I1, i2, e, o,e1,i22);
  input I1, i2, e;
  output o;
  input e1,i22;
//  output o1;
//  assign o = e1 ? i22 : 1'bz;
  assign o = I1;
  assign o =  e ? i2 : 1'bz; //warning on "o"
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1007: Tristate Buffer is not properly defined on "o".

Rule No 1008 Multiple Tri-state Cause Potential Bus Contention

Configurable Parameter

Argument type: none;

Description

Multiple drivers from tri-state buffer will possibly cause bus contention.

```
/////////////// Example : Document.v ///////////////////////

module test2 (sel1, sel2, a, b, out);
input sel1, sel2, a, b;
output out;
wire out; //warning here
assign out = sel1? a: 1'bz;
assign out = sel2? b: 1'bz;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1008: Multiple Tristate is not properly defined "out".
```

Rule No 1009 Parameter Name Case

Configurable Parameter

Argument type: Parameter Name case;

Description

Check to see if the parameters are all in upper (or lower) case.

```
/////////////// Example : Document.v ////////////////////////

module test (clock, reset, control, y);
input clock, reset, control;
output [2:0] y;
parameter st0 = 0; //warning on st0; ST0 is recommended
reg [1:0] current, next;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1009: Parameter "st0" cannot be in LOWER CASE.
```

Rule No 1010 Parameter Bit-width Too Long

Configurable Parameter

Argument type: none;

Description

Check to see if the bit width of the parameter exceeds 'length'

```
//////////////// Example: Document.v //////////////////////////

module test;
  parameter TW= 33'b100010000000000000000000000000010; //warning here
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1010: Generic Bit-width Too Long "33" for parameter "TW".
```

Rule No 1011 Reset is Driven by a Path with Potential Glitch

Configurable Parameter

Argument type: none;

Description

Check to see if there is any reset signal driven by a path with potential glitch.

```
/////////////// Example : Document.v ////////////////////////

module test (q, clk, en, reset, d);
  output q;
  input clk, en, reset, d;
  reg q;
  wire clk, en, reset, d;
  wire rst_en;
  and U_and_1(rst_en, reset, en); //warning on "rst_en"
  always @( posedge clk or negedge rst_en )
    if ( ~rst_en )
      q <= 1'b0;
    else
      q <= d;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1011: the set/reset signal should not be driven by a path "U_and_1<0>" with potential glitch.
```

Rule No 1012 No Output Port

Configurable Parameter

Argument type: none;

Description

Check there is any output port or not

```
//////////////// Example : Document.v //////////////////////////

module test (bind, b, c); //no output for module test
  input bind, b, c;
  and an(bind,b,c);
  //assign a = b;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1012: No Output Port is on "test.bind".
```

Rule No 1013 Gated Clock

Configurable Parameter

Argument type: none;

Description

The rule checks whether there is any gated clock in the design.

```
//////////////// Example : Document.v //////////////////////////

module test (q, clk, en, reset, d);
  output q;
  input clk, en, reset, d;
  reg q;
  wire clk, en, reset, d;
  wire clk_en;
  and U_and_1(clk_en, clk, en);//warning
  always @( posedge clk_en)//or negedge reset )
    if ( ~reset )
      q <= 1'b0;
    else
      q <= d;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1013: Gated Clock is on "U_and_1<0>".
```

Rule No 1014 Inverted Clock

Configurable Parameter

Argument type: none;

Description

Check to see if there is any inverted clock in the design.

```
//////////////// Example : Document.v //////////////////////////

module test (q, clk, reset, d);
  output q;
  input clk, reset, d;
  reg q;
  wire clk, reset, d;
  wire clk_i;
  not U_buf_1(clk_i, clk); //warning on "clk_i", clk_i is derived
                           //by an inverter
  always @( posedge clk_i or negedge reset )
    if ( ~reset )
      q <= 1'b0;
    else
      q <= d;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1014: Inverted Clock is on "U_buf_1<0>".
```

Rule No 1015 Buffered Clock

Configurable Parameter

Argument type: none;

Description

Check to see if there is any explicit buffered clock in the design.

```
/////////////// Example : Document.v ////////////////////////

module test (q, clk, reset, d);
  output q;
  input clk, reset, d;
  reg q;
  wire clk, reset, d;
  wire clk_i;
  buf U_buf_1(clk_i, clk);//warning on "clk_i", clk_i is drived
                        //by a buffer
  always @( posedge clk_i or negedge reset )
    if (~reset )
      q <= 1'b0;
    else
      q <= d;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1015: Buffered Clock is on "U_buf_1<0>".
```

Rule No 1016 Set Driven by Combinational Logic

Configurable Parameter

Argument type: none;

Description

Check to see if there is any set signal driven by a combinational logic.

```
/////////////// Example : Document.v ////////////////////////

module test (q, clk, en, set, d);
  output q;
  input clk, en, set, d;
  reg q;
  wire clk, en, set, d;
  wire set_en;
  parameter P = 2;
  and U_and_1(set_en, set, en); //warning on "set_en", is gated
  always @( posedge clk or negedge set_en )
    if (~set_en )
      q <= 1'b1;
    else
      q <= d;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1016: Violation on Set Gate "U_and_1<0>".
```

Rule No 1017 Reset Driven by Combinational Logic

Configurable Parameter

Argument type: none;

Description

Check to see if there is any reset signal driven by a combinational logic.

```
//////////////// Example : Document.v //////////////////////////

module test (q, clk, en, reset, d);
  output q;
  input clk, en, reset, d;
  reg q;
  wire clk, en, reset, d;
  wire rst_en;
  and U_and_1(rst_en, reset, en);//warning on "rst_en", is gated
  always @( posedge clk or negedge rst_en )
//  always @(clk | rst_en )
    if (rst_en )
      q <= 1'b0;
    else
      q <= d;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1017: Violation on Reset Gate "U_and_1<0>".
```

Rule No 1018 Line Too Long

Configurable Parameter

Argument type: none;

Description

Check to see if the line length exceeds 'length' characters.

```
//////////////// Example : Document.v //////////////////////////

module test;
        reg sel;
        reg [7:0] a,b,c,d,e,f,g;
        initial
        begin//:B
$monitor($time,,,,,,,"sel=%d,a=%d,b=%d,c=%d,d=%d,e=%d,f=%d,g=%d",sel,a,b,c,d,e,f,g);
     //do not type a line so long.
        a=12;
        b=13;
        c=14;
        e=15;
        f=16;
        g=17;
        sel=1;
        #200 $finish;
      end
      always
      begin//:A
        # 20 sel = ~sel;
      end
    endmodule
```

Run Command :

iverilog-0.8 -tlint Document.v

Linting Tool reports:

Violated 1018: Line Too Long.

Rule No 1019 Multi-bit Expression when One Bit Expression is Expected

Configurable Parameter

Argument type: none;

Description

Check to see if there is any condition expression is wider than single-bit.

```
/////////////// Example : Document.v ////////////////////////

module test (a, c);
  input [1:0] a;
  output c;
  reg c;
  always @(a)
    begin
    if (a) //warning here, one bit expression expected,
            //"a != 2'b00" is desired
      c = 1;
    else
      c = 0;
    end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

Violated 1019: More Than Expected Bit 2'b.

Rule No 1020 Logical or Bitwise OR Used in Event Control

Configurable Parameter

Argument type: none;

Description

Check to see if '|' or '||' is used in an event expression. The correct usage is 'or'.

```
/////////////// Example : Document.v ////////////////////////

module test (clk, rst, set, Data, q1, q2);
  input clk, rst, set;
  input Data;
  output [1:0] q1, q2;
  reg [1:0] q1, q2;
  //should be always@(posedge clk or posedge rst)
  always@(clk | rst)
    begin
      if (rst)
        q1 <= 0;
      else
        q1 <= Data;
    end
  //should be @(posedge clk or posedge set)
  always@(clk || set)
    begin
      if (set)
        q2 <= 1;
      else
        q2 <= Data;
    end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1020: 'or' (rather than |, ||) should be used in event expression "(clk....)".
Violated 1020: 'or' (rather than |, ||) should be used in event expression "(clk....)".
```

Rule No 1021 Drive Strength Not Synthesizable

Configurable Parameter

Argument type: none;

Description

Check to see if there is any drive strength because it cannot be synthesized.

//////////////// Example : Document.v //////////////////////////

```
module test (a, b);
  output a;
  input b;
  assign (strong1,pull0) b = a; //"strong1,pull0" non-synthesizable
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1021: drive strength "(pull)" should not be used because it is not synthesizable.
Violated 1021: drive strength "(strong)" should not be used because it is not synthesizable.
```

Rule No 1022 Unassigned in Data Clause

Configurable Parameter

Argument type: none;

Description

A register which has reset/set clause shall also be assigned in "data" clause.

```
//////////////// Example : Document.v //////////////////////////

module test (counter, clock, reset);
  input clock, reset;
  output [3:0] counter;
  reg [3:0] counter;
  wire [3:0] tmp;
  always @(posedge clock or negedge reset)
    if ( ~reset )
      counter = 0; //warning here
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

Violated 1022: register "counter<0>" should be assigned in data clause.

Rule No 1023 Special Type Port Connected to an Expression

Configurable Parameter

Argument type: none;

Description

Check to see if any special type port (TRI_ENABLE) connect to an expression.

```
/////////////// Example : Document.v ////////////////////////

module test ( a, b, c, y, d );
 input a, b, c, y;
 output d;
 reg d;
 always @(a or b or c or y)
   if(a&b|c)       // warning here if TRI_ENABLE is selected
     d = y;
   else
     d = 1'bz;
 endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1023: tri-state enable port should not be connect to an expression.

Rule No 1024 UDP Instance Not Synthesizable

Configurable Parameter

Argument type: none;

Description

Check to see if there is any UDP instance because it cannot be synthesized.

```
//////////////// Example : Document.v ///////////////////////////

module top;
  reg a, b, cin;
  wire sum;
  test u_test_0(sum, cin, a, b); //non-synthesizable, warning
endmodule
primitive test(sum, cin, a, b);
  output sum;
  input cin,a,b;
  table
  0 0 0 : 0;
  0 0 1 : 1;
  0 1 0 : 1;
  0 1 1 : 0;
  1 0 0 : 1;
  1 0 1 : 0;
  1 1 0 : 0;
  1 1 1 : 1;
  endtable
endprimitive
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1024: UDP instance "u_test_0" should not be used because it is not synthesizable.
```

Rule No 1025 Signal Driven by Both Blocking and Non-blocking Assignments

Configurable Parameter

Argument type: none;

Description

The rule checks whether there are any signals assigned by both blocking and non-blocking statements.

```
//////////////// Example : Document.v //////////////////////////

module test (c, a, b, en);
  input a, b, en;
  output c;
  reg c;
  always @(a or b)
    if (~en)
      c = a; //blocking assignment on "c"
    else
      c <= #2 b; //non-blocking assignment on "c"
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1025: signal "c" is assigned by both blocking and non-blocking assignments.
```

Rule No 1026 x/z Used in Case Label

Configurable Parameter

Argument type: none;

Description

The rule checks whether there is any x, z used in case label of case statements.

```
/////////////// Example : Document.v ////////////////////////

module test (out, clk, in, cas);
  input clk, in, cas;
  output out;
  reg out;
  always @(clk)
    case (cas)
      1'b0: out = !in;
      1'b1: out = in;
      1'bz: out = 1; //warning on "1'bz"
      1'bx: out = 0; //warning on "1'bx"
      default: out = 1;
    endcase
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1026: Violation "z" Used in Case Label.
Violated 1026: Violation "z" Used in Case Label.
```

Rule No 1027 Casex or Casez Detected

Configurable Parameter

Argument type: none;

Description

check to see if there is any casex or casez statement used

```
/////////////// Example : Document.v ////////////////////////

module test (out0,in1,in2,in3,sel);
  input [1:0] in1,in2,in3,sel;
  output [1:0] out0;
  reg [1:0] out0;
  always @( in1 or in2 or sel) begin
    casex(sel) //warning here
      2'b00: out0 = in1;
      2'b01: out0 = in2;
      2'b10: out0 = in3;
      default: out0 = in1;
    endcase
  end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1027: "casex" statement Detected.
```

Rule No 1028 Casex(z) DontCare

Configurable Parameter

Argument type: none;

Description

complex casex or casez decreases circuit quality with don't care

```
//////////////// Example: Document.v /////////////////////////

module test (out0,in1,in2,in3,sel);
  input [1:0] in1,in2,in3,sel;
  output [1:0] out0;
  reg [1:0] out0;
  always @( in1 or in2 or sel) begin
    casex(sel) //warning here
      2'b00: out0 = in1;
      2'b01: out0 = in2;
      2'b10: out0 = in3;
      default: out0 = in1;
    endcase
  end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1028: complex "casex" decreases circuit quality with.
```

Rule No 1029 Blocking/Non-blocking Assignment in Edge-triggered Block

Configurable Parameter

Argument type: none;

Description

The rule checks whether there are any blocking or non-blocking assignments used in an edge-triggered block.

```
//////////////// Example : Document.v //////////////////////////

module test;
  reg clock;
  reg a,b,c;
  always @(posedge clock)
    begin
      a = b;
      c <= a; //choose BLOCKING, IGNORE_DEPEND;
             //block assignment in edge-trigger block will cause
             //mismatch between pre-synthesis and
             //post-synthesis simulation
    end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1029: "BLOCKING" assignment is on "a" used in an edge triggered block.
Violated 1029: "NON-BLOCKING" assignment is on "c" used in an edge triggered block.
```

Rule No 1030 Vector Used in a Single-bit Logical Operation

Configurable Parameter

Argument type: none;

Description

The rule checks whether there are any vectors used as operands for a single-bit logical operator (&&, || or !).

```
/////////////// Example : Document.v ////////////////////////

module test (.pc(c), .pa(a), .pb(b));
  input [2:0] a, b;
  output [2:0] c;
  reg [2:0] c;
  always @(a or b)
    begin
    if (a && b) //warning on "a" and "b"
      c = 1;
    if (!(a || b)) //warning on "a" and "b"
      c = 0;
    end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1030: "vector" "b" is used in a single-bit logical operation.
Violated 1030: "vector" "a" is used in a single-bit logical operation.
```

Rule No 1031 Signal Used as Synchronous and Asynchronous Reset

Configurable Parameter

Argument type: none;

Description

Check to see if there is any signal that is used as synchronous and asynchronous reset scimultaneously.

```
/////////////// Example : Document.v ////////////////////////

module test (y1, y2, data, clock, preset, clear);
  input data, clock, preset, clear;
  output y1, y2;
  reg y1, y2;
  always @(posedge clock or posedge clear or posedge preset)
            //asynchronous set signal "preset"
    begin: forset
    if (clear)
      y1 = 0;
    else
      if (preset)
         y1 = 1;
      else
         y1 = data;
    end
  always @(posedge clock)
    begin
    if (clear)
      y2 = 0;
    else
      if (preset)//synchronous set "preset", mixed
         y2 = 1;
      else
         y2 = data;
    end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1031: "Sync" Reset detected.

Rule No 1032 SystemVerilog Reserved Words

Configurable Parameter

Argument type: none;

Description

This rule checks whether any SystemVerilog reserved words are used as an identifier in a Verilog design. The SystemVerilog reserved words are listed below.
"alias", "always", "always_comb", "always_ff", "always_latch",
"and", "assert", "assign", "assume", "automatic",
"before", "begin", "bind", "bins", "binsof",
"bit", "break", "buf", "bufif0", "bufif1",
"byte", "case", "casex", "casez", "cell",
"chandle", "class", "clocking", "cmos", "config",
"const", "constraint", "context", "continue", "cover",
"covergroup", "coverpoint", "cross", "deassign", "default",
"defparam", "design", "disable", "dist", "do",
"edge", "else", "end", "endcase", "endclass",
"endclocking", "endconfig", "endfunction", "endgenerate", "endgroup",
"endinterface", "endmodule", "endpackage", "endprimitive", "endprogram",
"endproperty", "endspecify", "endsequence", "endtable", "endtask",
"enum", "event", "expect", "export", "extends",
"extern", "final", "first_match", "for", "force",
"foreach", "forever", "fork", "forkjoin", "function",
"generate", "genvar", "highz0", "highz1", "if",
"iff", "ifnone", "ignore_bins", "illegal_bins", "import",
"incdir", "include", "initial", "inout", "input",
"inside", "instance", "int", "integer", "interface",
"intersect", "join", "join_any", "join_none", "large",
"liblist", "library", "local", "localparam", "logic",
"longint", "macromodule", "matches", "medium", "modport",
"module", "nand", "negedge", "new", "nmos",
"nor", "noshowcancelled", "not", "notif0", "notif1",
"null", "or", "output", "package", "packed",
"parameter", "pmos", "posedge", "primitive", "priority",
"program", "property", "protected", "pull0", "pull1",
"pulldown", "pullup", "pulsestyle_onevent", "pulsestyle_ondetect", "pure",
"rand", "randc", "randcase", "randsequence", "rcmos",
"real", "realtime", "ref", "reg", "release",
"repeat", "return", "rnmos", "rpmos", "rtran",
"rtranif0", "rtranif1", "scalared", "sequence", "shortint",
"shortreal", "showcancelled", "signed", "small", "solve",
"specify", "specparam", "static", "string", "strong0",
"strong1", "struct", "super", "supply0", "supply1",
"table", "tagged", "task", "this", "throughout",
"time", "timeprecision", "timeunit", "tran", "tranif0",
"tranif1", "tri", "tri0", "tri1", "triand",
"trior", "trireg", "type", "typedef", "union",

"unique", "unsigned", "use", "var", "vectored",
"virtual", "void", "wait", "wait_order", "wand",
"weak0", "weak1", "while", "wildcard", "wire",
"with", "within", "wor", "xnor", "xor".

```
/////////////// Example : Document.v ////////////////////////

module test;
wire bind,b,c;
and AA(bind,b,c);
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1032: System Verilog Reserved Words "bind".
```

Rule No 1033 Verilog Reserved Words

Configurable Parameter

Argument type: none;

Description

Do not use Verilog reserved words to avoid translation error Verilog reserved words: "always", "and", "assign", "begin", "buf", "bufif0", "bufif1", "case", "casex", "casez", "cmos", "deassign", "default", "defparam", "disable", "edge", "else", "end",? "endcase", "endmodule", "endfunction", "endprimitive", "endspecify", "endtable", "endtask", "event", "for", "force", "forever", "fork", "function", "highz0", "highz1", "if", "ifnono", "initial", "inout", "input", "integer", "join", "large", "macromodule", "medium", "module", "nand", "negedge", "nmos", "nor", "not", "notif0", "notif1", "or", "output", "parameter", "pmos", "posedge", "primitive", "pull0", "pull1", "pullup", "pulldown", "rcmos", "real", "realtime", "reg", "release", "repeat", "rnmos", "rpmos", "rtran", "rtranif0", "rtranif1", "scalared", "small", "specify", "specparam", "strong0", "strong1", "supply0", "sypply1", "table", "task", "time", "tran", "tranif0", "tranif1", "tri", "tri0","tri1", "triand", "trior", "trireg", "vectored", "wait", "wand", "weak0", "weak1", "while", "wire","wor", "xnor", "xor"

```
/////////////// Example : Document.v ////////////////////////

module test;
wire wor_o,b,c;
and AA(wor_o,b,c);
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1033: Keyword "wor_o" is part of Verilog Reserved Word.
```

Rule No 1034 VHDL Reserved Words

Configurable Parameter

Argument type: none;

Description

Check to see if there is any VHDL reserved word used. VHDL reserved words: "abs", "access", "after", "alias", "all", "and", "architecture", "array", "assert", "attribute", "begin", "block","body", "buffer", "bus", "case", "component", "configuration", "constant", "disconnect", "downto","else", "elsif", "end", "entity", "exit", "file", "for", "function", "generate", "generic", "group", "guarded", "if", "impure", "in", "inertial", "inout", "is", "label", "library", "linkage", "literal","loop", "map", "mod", "nand", "new", "next", "nor", "not", "null", "of", "on", "open", "or","others", "out", "package", "port", "postponed", "procedure", "process", "pure", "range", "record","register", "reject", "rem", "report", "return", "rol", "ror", "select", "severity", "shared", "signal","sla", "sll", "sra", "srl", "subtype", "then", "to", "transport", "type", "unaffected", "units", "until","use", "variable", "wait", "when", "while", "with", "xnor", "xor"

```
/////////////// Example : Document.v ////////////////////////

module test;
wire in,b,c;
and AA(in,b,c);
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1034: VHDL Reserved Words "in".

Rule No 1035 Insufficient Index Variable

Configurable Parameter

Argument type: none;

Description

The rule checks whether index signals are wide enough to address all the bits in the indexed vector signals.

```
//////////////// Example: Document.v //////////////////////////

module test (a, clk, num, b);
  input clk;
  input [7:0] a;
  input [1:0] num;
  output b;
  reg b;
  always @(posedge clk)
    b = a[num]; //warning
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1035: Insufficient Index "2".
```

Rule No 1036 Oversized Index Variable

Configurable Parameter

Argument type: none;

Description

The rule checks whether the index signal exceeds the range of bus signals or multiple dimensional arrays.

```
//////////////// Example : Document.v //////////////////////////

module test(a, clk, num, b);
  input clk;
  input [3:0] a;
  input [4:0] num;
  output b;
  reg b;
  always @(posedge clk)
    b = a[num]; //warning on "a[num]"
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1036: Oversized Index "5".
```

Rule No 1037 Inout as Input Only in Shift

Configurable Parameter

Argument type: none;

Description

Check to see whether inout port be input only in shift mode.

```
//////////////// Example : Document.v /////////////////////////

module test(io,data,clk,set,reset,en);
inout io;
input data,clk,set,reset,en;
reg q;
wire d;
assign d=data&io;
always @(posedge clk)
  if(~reset)
     q='b0;
  else if(set)
     q='b1;
  else
     q=d;
assign io= (en?q:'bz);
endmodule
```

Run Command :

iverilog-0.8 -tlint Document.v

Linting Tool reports:

Violated 1037: inout port "io" should be input only in shift.

Rule No 1038 Tri-State Disabled in Shift

Configurable Parameter

Argument type: none;

Description

Check to see whether tri-state be disabled in shift mode.

```
//////////////// Example : Document.v //////////////////////////

module test(io,data,clk,set,reset,en);
inout io;
input data,clk,set,reset,en;
reg q;
wire d;
assign d=data&io;
always @(posedge clk)
  if(~reset)
     q='b0;
  else if(set)
     q='b1;
  else
     q=d;
assign io= (en?q:'bz);
endmodule
```

Run Command :

iverilog-0.8 -tlint Document.v

Linting Tool reports:

Violated 1038: Tri-state "io" should be disabled in shift.

Rule No 1039 Library Cell Instantiated

Configurable Parameter

Argument type: none;

Description

The rule checks whether any library cells instantiated in the design.

```
/////////////// Example : Document.v ////////////////////////

module smp(dout,ck,res,din);
input ck,res,din;
output dout;
reg    douti;
parameter D1 = 1;
always @(posedge ck or negedge res)
       if ( ~res )
          douti <= #D1 1'b0;
       else
          douti <= #D1 din;
JANIV OBUF (.A(douti), .Z(dout));
endmodule

`celldefine
module JANIV(A, Z);
input A;
output Z;
assign Z = A;
endmodule
`endcelldefine
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1039: library cell "JANIV" is instantiated.
```

Rule No 1040 No Falling Active Clock Used

Configurable Parameter

Argument type: none;

Description

Check to see if there is any falling active clock used.

```
//////////////// Example : Document.v //////////////////////////

module dffn( clk, d, q );
  input clk, d;
  output q;
  reg q;
  always @( negedge clk ) //warning here
    q <= d;
endmodule
module test( clock, d, q );
  input clock;
  input [3:0] d;
  output [3:0] q;
  wire clkn;
  assign clkn = ~clock;
  dffn i_dffn_1( clkn, d[0], q[0] );
  dffn i_dffn_2( clock, d[1], q[1] );
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1040: falling active clock "clkn" should not be used.
Violated 1040: falling active clock "clock" should not be used.
```

Rule No 1041 Clock Active on Both Edges

Configurable Parameter

Argument type: none;

Description

Check to see if a clock source triggers flip-flops on both rising and falling edges.

```
//////////////// Example : Document.v /////////////////////////

module test(clk, d, q);
input clk, d;
output q;
wire clk, d;
reg q1, q;
always @(posedge clk)
  q1 = d;
always @(negedge clk)
  q = q1;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1041: clock source "clk" should not trigger flip-flops on both rising and falling edges.
```

Rule No 1042 Parameter Name Prefix or Suffix

Configurable Parameter

Argument type: none;

Description

Check to see if the parameter name has a recommended prefix or suffix.

```
//////////////// Example : Document.v //////////////////////////

module test (clock, reset, control, y);
  input clock, reset, control;
  output [2:0] y;
  reg [2:0] y;
  parameter st0 = 0; //warning on st0, p_st0 is recommended
  reg [1:0] current, next;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1042: parameter name "st0" does not match to regular expression p_.*.

Rule No 1043 Parameter Name Length

Configurable Parameter

Argument type: none;

Description

Check to see if the length of the parameter name is in the specified range

```
/////////////// Example : Document.v ////////////////////////

module test;
  parameter par_012345678901234 = 5;
           //warning on "par_012345678901234"
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1043: The length of parameter "par_012345678901234" is unconventional and should be in the range from 3 to 16.

Rule No 1044 Case Label out of Boundary

Configurable Parameter

Argument type: none;

Description

The rule checks whether there is a case label whose value exceeds the boundary of case variable.

```
/////////////// Example : Document.v ////////////////////////

module test (Clock);
  input Clock;
  reg [3:0] state;
  reg [2:0] data;
  always@(Clock)
    begin
      case (state)
        16: data <= 3'b111; //warning
        15: data <= 3'b110;
        default: data <= 3'b000;
      endcase
    end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1044: violation as case label 000010j exceed "5'b"
```

Rule No 1045 Different Bits of Vector Driven in Different Blocks

Configurable Parameter

Argument type: none;

Description

Check to see if there are different bits of signal that is driven in different blocks.

```
//////////////// Example : Document.v //////////////////////////

module test (a, b, c);
  input a, b;
  output [2:0] c;
  reg [2:0] c;
  initial
      c[2] = 1;
  always@(a)
     c[0]=a;           //warning here
  always@(b)
     c[1]=b;           //warning here, "c[1]" and "c[0]" are used
                      //in different always blocks
endmodule
```

Run Command :

iverilog-0.8 -tlint Document.v

Linting Tool reports:

Violated 1045: different bits of signal "c[0]" and "c[1]" should not be driven in different blocks.

Rule No 1046 Default is Not Found

Configurable Parameter

Argument type: none;

Description

Check to see if there is any default clause in a full case statement.

```
/////////////// Example : Document.v ////////////////////////

module test (sel,a,c);
  input [1:0] a ;
  input [1:0] sel;
  output [1:0] c;
  reg [1:0] c;
  `define ZERO 0
  `define ONE 1
  `define TWO 2
  `define THREE 3
  always @ (a or sel)
    case (sel)    //warning here, default item is lost
                  //If sel signal is X or Z , it may cause problem
    `ZERO: c = 2'b00;
    `ONE : c = 2'b01;
    `TWO : c = a;
     //default: c= 2'b11;
    //`THREE: c= 2'b11;
    endcase
endmodule
```

Run Command :

iverilog-0.8 -tlint Document.v

Linting Tool reports:

Violated 1046: Default Not Found.

Rule No 1047 Casez Label Contains X

Configurable Parameter

Argument type: none;

Description

Check to see if there is any casez item that contains x

```
//////////////// Example : Document.v /////////////////////////

module test (Clock);
  input Clock;
  reg state;
  reg out;
  always@(Clock)
    begin
      casez (state)
        1'bz: out = 1;
        1'bx: out = 0; //warning on "1'bx"
        default: out = 1;
      endcase
    end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1047: Violation x Used in Casez Label.
```

Rule No 1048 Write Enable Signals for Memories Should be Disabled in the Test Mode

Configurable Parameter

Argument type: none;

Description

write enable signals for memories should be disabled in the test mode.

```
/////////////// Example : Document.v ////////////////////////

module test(test_mode, WE, WE_mem);
  input test_mode;
  input WE;
  output WE_mem;
  assign WE_mem = test_mode | WE;
  //code inferred to memory operation
  //...
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1048: write enable signal "test.WE_mem" for memories should be disabled in the test mode.
```

Rule No 1049 Case-like If Statement

Configurable Parameter

Argument type: none;

Description

Check to see if cascaded if-else statement should be re-written using a case statement.

```
//////////////// Example : Document.v ///////////////////////////

module test (out, sel, in);
  parameter SEL_WIDTH = 2, D_WIDTH = 4;
  output out;
  input [SEL_WIDTH-1:0] sel;
  input [D_WIDTH-1:0] in;
  reg out;
  always //@(sel or in)
    if (sel == 0) //this "if" statement should be replaced with
                   //case statement, which is parallel process;
                   //warning
      out = in[0];
    else
      if (sel == 1)
         out = in[1];
      else
         if (sel == 2)
           out = in[2];
         else
           out = in[3];
endmodule
```

Run Command :

iverilog-0.8 -tlint Document.v

Linting Tool reports:

Violated 1049: Case Like If Else.

Rule No 1050 Z or X Used in Conditional Expression

Configurable Parameter

Argument type: none;

Description

The rule checks whether 'x' or 'z' is used in any conditional expressions.

```
/////////////// Example : Document.v ////////////////////////

module test (dataout, s, datain,clk);
 input clk;
  parameter WIDTH = 4;
  output dataout;
  input s;
  input [WIDTH-1:0] datain;
  reg dataout;
  always @(clk)//s or datain)
    begin
    if (s == 'bz) //warning
      dataout = datain[0];
    else
      if (s == 'bx) //warning
         dataout = datain[1];
      else
  //       if (s == 'b0)
  //         dataout = datain[2];
    //     else
           dataout = datain[3];
    end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1050: Z or X Used in Conditional Expression pin2 of "s".
Violated 1050: Z or X Used in Conditional Expression pin2 of "s".
```

Rule No 1051 Blocking and Non-blocking Statements in the Same Always Block

Configurable Parameter

Argument type: none;

Description

The rule checks whether there is an always block containing both blocking and non-blocking statements.

```
//////////////// Example : Document.v //////////////////////////

module test (c, a, b, en);
  input a, b, en;
  output c;
  reg c;
  always @(a or b)
    if (~en)
      c = a; //blocking assignment on "c"
    else
      c <= #2 b; //non-blocking assignment on "c"
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

Violated 1051: Blocking and Non-blocking Statements in the Same Always Block.

Rule No 1052 Delay Control Ignored by Synthesis

Configurable Parameter

Argument type: none;

Description

Check to see if there is any delay which may cause difference between simulation

result of pre-synthesis and post-synthesis.

```
//////////////// Example : Document.v //////////////////////////

module test (clk, data, y);
  input clk, data;
  output y;
  reg y;
  always @(posedge clk)
    #10 y = data;//"#10" ignored by synthesis, warning
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1052:     "#10" Delay Control Ignored by Synthesis.
```

Rule No 1053 System Task Call Not Synthesizable

Configurable Parameter

Argument type: none;

Description

Check to see if there is any system task call statement because it cannot be synthesized.

```
/////////////// Example : Document.v ////////////////////////

module init;
        reg clk, rst, in;
        wire out;

        initial begin
         $monitor($time,,,"clk=%d,rst=%b,in=%b,out=%b",clk,rst,in,out);
         clk = 0;
         rst = 1;
         in = 0;
         #2 rst = 0;
         #10 rst = 1;
         #50 $finish;            //warning here
       end
       always
         #4 clk = !clk;
       always
        #10 in = !in;
    endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1053: System Task Call "$monitor;" Not Synthesizable.
Violated 1053: System Task Call "$finish;" Not Synthesizable.
```

Rule No 1054 Force Statement Not Synthesizable

Configurable Parameter

Argument type: none;

Description

Check to see if there is any force statement because it cannot be synthesized.

```
/////////////// Example : Document.v ////////////////////////

module test;
  reg a,b,c,d,e;
  initial
  begin
    $monitor($time,,,"a=%d,b=%d,c=%d,d=%d,e=%d",a,b,c,d,e);
    assign d = a & b & c;
    a = 1;
    b = 0;
    c = 1;
    force d = (a | b | c); //"force" non-synthesizable, warning
    force e = (a | b | c); //"force" non-synthesizable, warning
    release d;
    release e;
    #10 $finish;
  end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1054: Force Statement Not Synthesizable.
Violated 1054: Force Statement Not Synthesizable.
```

Rule No 1055 Matching Release Statement Not Found

Configurable Parameter

Argument type: none;

Description

Check to see if there is any force statement that has no corresponding release statement.

```
//////////////// Example : Document.v //////////////////////////

module top;
  reg a, b, c, d;
  wire e;
  initial
  begin
    assign d = a & b & c;
    a = 1;
    b = 0;
    c = 1;
    #10;
    force d = (a | b | c); //"d" is forced but never been
                           //released, warning
    force e = (a | b | c); //"e" is forced but never been
                           //released, warning
    #10 $finish;
  end
  test u_test_0(e, a, b, c);
endmodule
module test (e,a,b,c);
  output e;
  input a,b,c;
  wire e;
  and and1(e,a,b,c);
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1055: Matching Release Not Found.
Violated 1055: Matching Release Not Found.
```

Rule No 1056 Release Statement Not Synthesizable

Configurable Parameter

Argument type: none;

Description

Check to see if there is any release statement because it cannot be synthesized.

```
/////////////// Example : Document.v ////////////////////////

module test;
  reg a, b, c, d, e;
  initial
  begin
    $monitor($time,,,"a=%d,b=%d,c=%d,d=%d,e=%d",a,b,c,d,e);
    assign d = a & b & c;
    a = 1;
    b = 0;
    c = 1;
    force d = (a | b | c);
    force e = (a | b | c);
    release d; //"release" non-synthesizable, warning
    release e; //"release" non-synthesizable, warning
    #10 $finish;
  end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1056: Release Statement Not Synthesizable.
Violated 1056: Release Statement Not Synthesizable.
```

Rule No 1057 Matching Force Statement Not Found

Configurable Parameter

Argument type: none;

Description

Check to see if there is any release statement that has no corresponding force statement.

```
/////////////// Example : Document.v ////////////////////////

module top;
  reg a, b, c, d;
  wire e;
  initial
  begin
    assign d = a & b & c;
    a = 1;
    b = 0;
    c = 1;
    #10;
    release d; //"d" is released but never been forced, warning
    release e; //"e" is released but never been forced, warning
    #10 $finish;
  end
  test u_test_0(e, a, b, c);
endmodule
module test (e,a,b,c);
  output e;
  input a,b,c;
  wire e;
  and and1(e,a,b,c);
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1057: Matching Force Not Found.
Violated 1057: Matching Force Not Found.
```

Rule No 1058 Signal Has Never Been Referenced Assigned.

Configurable Parameter

Argument type: none;

Description

Check to see if there is any signal that has never been Referenced.

```
/////////////// Example : Document.v ////////////////////////

module test;
  wire c;
  reg a,b,d;
  initial
  begin
  assign a=b;
  assign d=c;
  end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1058: Signal "b" Has Never Been Assigned.
Violated 1058: Signal "c" Has Never Been Assigned.
```

Rule No 1059 Signal Has Never Been Referenced referenced.

Configurable Parameter

Argument type: none;

Description

Check to see if there is any signal that has never been referenced.

```
//////////////// Example : Document.v //////////////////////////

module test;
  wire c;
  reg a,b,d;
  initial
  begin
  assign a=b;
  assign d=c;
  end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1059: Signal "a" Has Never Been Refferenced.
Violated 1059: Signal "d" Has Never Been Refferenced.
```

Rule No 1060 While Statement Not Synthesizable

Configurable Parameter

Argument type: none;

Description

Check to see if there is any while statement because it cannot be synthesized.

```
/////////////// Example : Document.v ////////////////////////

module test (clk,a, b, y);
  input clk;
  input [1:0] a, b;
  output [3:0] y;
  reg [3:0] y;
  integer i;
  always @(clk)
    begin
    i = 0;
    while (i <= 3)//"while", warning
      begin
      y[i] = a[i];
      i = i + 1;
      end
    end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

Violated 1060: While/for Statement Not Synthesizable.

Rule No 1061 Matching Deassign Statement Not Found

Configurable Parameter

Argument type: none;

Description

Check to see if there is any assign statement that has no corresponding deassign statement.

```
//////////////// Example : Document.v //////////////////////////

module testini;
  reg [1:0] a, b, temp;
  wire [1:0] c;
  initial
  begin
    a = 2'b01;
    b = 2'b11;
    #50 a <= ~a;
    assign temp=a; //"temp" is deassigned but never been assigned,
                   //warning
    #100 $finish;
  end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

Violated 1061: Matching Dassign Statement Not Found.

Rule No 1062 Matching Assign Statement Not Found

Configurable Parameter

Argument type: none;

Description

Check to see if there is any deassign statement that has no corresponding assign statement.

```
/////////////// Example : Document.v ////////////////////////

module testini;
  reg [1:0] a, b, temp;
  wire [1:0] c;
  initial
  begin
    a = 2'b01;
    b = 2'b11;
    #50 a <= ~a;
    deassign temp; //"temp" is deassigned but never been assigned,
                   //warning
    #100 $finish;
  end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

Violated 1062: Matching Assign Statement Not Found.

Rule No 1063 Signal Has Been Assigned in More

Configurable Parameter

Argument type: none;

Description

Check to see if there is any signal that has been assigned in more than one blocks.

```
/////////////// Example : Document.v ////////////////////////

module test;
  reg a, b, c;
  initial begin
    a=b;
  end
  initial begin
    a=c; //"a" has been assigned in more than one block, warning
  end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1063: Signal "a" Assigned more than one Block.
```

Rule No 1064 Logical/Arithmetic/BitWise Operation in Case Selection Expression

Configurable Parameter

Argument type: none;

Description

This rule checks whether there are any logical, arithmetic or bitwise operation used in the case selection expression.

```
//////////////// Example : Document.v //////////////////////////

module smp(zo,sel1,sel2,clk);
 input clk;
 input[1:0] sel1,sel2;
 output[3:0] zo;
 reg [3:0]  zo;
 always@(clk)
    case(sel1&sel2)
     2'b00:     zo = 4'b0001;
     2'b01:     zo = 4'b0010;
     2'b10:     zo = 4'b0100;
     2'b11:     zo = 4'b1000;
     default: zo = 4'b0000;
    endcase
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1064: Some operation "&" in Case Selection Expression.
```

Rule No 1065 Nested Edge-triggers

Configurable Parameter

Argument type: none;

Description

Check to see if there is any nested edge-triggered construct.

```
//////////////// Example : Document.v /////////////////////////

module initval;
       reg clock,reset;
        wire [8:0] count,y;

        initial
          begin

            $monitor($time,,,"clock=%d,count=%d,reset=%d,y=%d",clock,count,reset,y);
            clock=0;
            reset=0;
           #20 reset=1;
           #10 reset=0;
           #50 $finish;
         end
       always
         #4 clock=!clock;

       test cc (y,clock,reset,count);
     endmodule
```

Run Command :

iverilog-0.8 -tlint Document.v

Linting Tool reports:

Violated 1065: nested edge-triggered constructs posedge clock should not be used.

```
//////////////// Example : Document.v ///////////////////////////

     module test (y,clock,reset,count);
       input clock,reset;
       output [8:0] count,y;
       reg [8:0] count,y;

       initial
         count<=0;
       always @(posedge clock)
         begin
           if ( reset )
              count = 0;
          else
              count= count + 1;
           @(posedge clock)             //warning here
              y=count;
         end
     endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1065: nested edge-triggered constructs posedge clock should not be used.
```

Rule No 1066 Undefined Repeat Expression

Configurable Parameter

Argument type: none;

Description

Check to see if there is any repeat expression that is evaluated to X or Z.

```
//////////////// Example : Document.v //////////////////////////

module test (clk,b,c);
  input clk;
  input [7:0] b;
  output [7:0]c;
  reg [7:0]c;
  always @(clk) begin
    repeat(4'b0x10) //repeat zero times
      c=b;
  end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

Violated 1066: repeat expression evaluates to X or Z, causing it to repeat zero or unknown times.

Rule No 1067 Repeat Statement Not Synthesizable

Configurable Parameter

Argument type: none;

Description

Check to see if there is any repeat statement because it cannot be synthesized.

```
/////////////// Example : Document.v ////////////////////////

module test (y, c, d, a, b);
  input a, b;
  output c, d;
  output [1:0] y;
  reg c, d;
  reg [1:0] y;
  always @(a or b)
    begin
    repeat(3) {c, d} = {a, b};//"repeat" non-synthesizable, warning
    y = repeatfun(a, b);
    end
  function [1:0] repeatfun;
    input a, b;
    repeat(6) repeatfun = {b, a};//"repeat" non-synthesizable, warning
  endfunction
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

Violated 1067: repeat statement should not be used because it is not synthesizable.

Rule No 1068 Event Control in Unsuitable Place is Not Synthesizable

Configurable Parameter

Argument type: none;

Description

Check to see if there is any event control in unsuitable place.

```
//////////////// Example : Document.v //////////////////////////

module test (clk, data, y);
  input clk, data;
  output y;
  reg y;
  always @(posedge clk)
    @( data ) y = data;//"@(data)" non-synthesizable, warning
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

Violated 1068: event control in unsuitable place is not synthesizable.

Rule No 1069 Task Call Not Synthesizable

Configurable Parameter

Argument type: none;

Description

Check to see if there is any task call statement because it cannot be synthesized.

```
/////////////// Example : Document.v ////////////////////////

module test;
  reg clock;
  reg a;
  reg b;
  reg y;
  always @(posedge clock)
    multiply(a, b, y); //"multiply" non-synthesizable, warning
  task multiply;
    input a;
    input b;
    output y1;
    begin: serialMult
      reg c, d;
      c = a;
      d = b;
      y1 = 0;
      repeat (5)
         begin
           y1 = c & d;
         end
    end
  endtask
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

Violated 1069: task call statement multiply(...) should not be used because it is not synthesizable.

Rule No 1070 Disable Statement Not Synthesizable

Configurable Parameter

Argument type: none;

Description

Check to see if there is any disable statement because it cannot be synthesized.

```
//////////////// Example : Document.v /////////////////////////

module test;
  reg clock, reset;
  reg a;
  reg b;
  reg y;
  always @(posedge clock)
    multiply(a, b, y);
  always @( negedge reset )
    disable multiply; //"disable" non-synthesizable, warning here
  task multiply;
    input a;
    input b;
    output y1;
    begin: serialMult
      reg c, d;
      c = a;
      d = b;
      y1 = 0;
      repeat (5)
         begin
           y1 = c & d;
         end
    end
  endtask
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

Violated 1070: disable statement disable "multiply" should not be used because it is not synthesizable.

Rule No 1071 number not execced threshold value

Configurable Parameter

Argument type: none;

Description

The number of line in always construct should not be up to threshold value.

```
//////////////// Example : Document.v //////////////////////////

module test;
        reg sel;
        reg [7:0] a,b,c,d,e,f,g;
        initial
        begin//:B

        $monitor($time,,,,,,"sel=%d,a=%d,b=%d,c=%d,d=%d,e=%d,f=%d,g=%d",sel,a,b,c,d,e,f,g);
        //do not type a line so long.
        a=12;
        b=13;
        c=14;
        e=15;
        f=16;
        g=17;
        #2 sel=1;
        #4 sel=0;
        #6 sel=1;
        #200 $finish;
      end
      always
      begin//:A
        # 20 sel = ~sel;
      end
    endmodule
```

Run Command :

iverilog-0.8 -tlint Document.v

Linting Tool reports:

Violated 1071: number of line in block should not be up to 11 > 10.

Rule No 1072 Delay in Non-blocking Assignment

Configurable Parameter

Argument type: none;

Description

Check to see if there is any delay used in a non-blocking assignment.

```
/////////////// Example : Document.v ////////////////////////

module test (q, clock, reset, d);
  output q;
  input clock, reset, d;
  reg q;
  wire clock, reset, d;
  parameter D_RQ = 1, D_CQ = 2;
  always @(posedge clock or negedge reset)
    if (~reset)
      #D_RQ q <= 0; //delay control "#D_RQ" may cause
                  //non-blocking effect invalid, warning
    else
      #D_CQ q <= d; //delay control "#D_CQ" may cause
                  //non-blocking effect invalid, warning
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1072: delay should not be used in a "q" non-blocking assignment.
Violated 1072: delay should not be used in a "q" non-blocking assignment.
```

Rule No 1073 Blocking/Non-blocking Assignment in Combinational Block

Configurable Parameter

Argument type: none;

Description

The rule checks whether there are any blocking or non-blocking assignments used in combinational blocks.

```
/////////////// Example : Document.v ////////////////////////

module test;
  reg in;
  reg a,o;
  always @(in)
    begin
      a <= in;
      o <= a; //choose NONBLOCKING, IGNORE_DEPEND;
              //non-block assignment in combinational block will
              //cause mismatch between pre-synthesis and
              //post-synthesis simulation
    end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1073: NONBLOCKING assignment "a" is used in combinational block.
Violated 1073: NONBLOCKING assignment "o" is used in combinational block.
```

Rule No 1074 Negative Value Assignment

Configurable Parameter

Argument type: none;

Description

This rule checks whether there are any negative values used in assignments.

```
/////////////// Example: Document.v ////////////////////////

module smp(A,B,C,G,H,Z);
input A,B,C,G,H;
output Z;
reg     Z;
 always@( A or B or C or G or H )
    case ({G,H})
     2'b00 :   Z <= A;
     2'b01 :   Z <= B;
     2'b10 :   Z <= C;
     2'b11 :   Z <= -1;
     default : Z <= 1'bx;
    endcase
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1074: do not assign signal with negative value "-1"
```

Rule No 1075 Signal Assigned to Self

Configurable Parameter

Argument type: none;

Description

Check to see if there is any signal assigned to itself.

```
/////////////// Example : Document.v ////////////////////////

module test (out, in);
  output [3:0] out;
  input [3:0] in;
  reg [3:0] out;
  always @( in or out ) begin
    out[0] = out[0];       //warning on "out[0]"
    out[1] = out[1] & 1;   //warning on "out[1]"
    out[2] = out[2] | 'h0; //warning on "out[2]"
    out[3] = in[3];
  end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1075: signal "out<0>" should not be assigned to itself.
Violated 1075: signal "out<1>" should not be assigned to itself.
Violated 1075: signal "out<2>" should not be assigned to itself.
```

Rule No 1076 Fixed Value of Case Selection Expression

Configurable Parameter

Argument type: none;

Description

This rule checks whether case selection expression is a fixed value.

```
/////////////// Example : Document.v ////////////////////////

module smp(zo,sel);
 input[3:0] sel;
 output[3:0] zo;
 reg [3:0]   zo;
 always@( sel )
    case(1'b1)
     sel[0]:     zo = 4'b0001;
     sel[1]:     zo = 4'b0010;
     sel[2]:     zo = 4'b0100;
     sel[3]:     zo = 4'b1000;
     default:    zo = 4'b0000;
    endcase
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1076: the value of case selection expression "1!j" is fixed.
```

Rule No 1077 Variable Updated Twice in Same Time Point

Configurable Parameter

Argument type: none;

Description

Check to see if there is any variable assigned in two assignments at the same time point.

```
/////////////// Example : Document.v ////////////////////////

module block( clk, a, b);
input clk, b;
output a;
reg a;
always @(posedge clk) begin
  a <= 1;
  if (b)
    a <= 0;
end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1077: variable "a" is assigned in another assignment in same time point.
```

Rule No 1078 Empty Process

Configurable Parameter

Argument type: none;

Description

The rule checks whether there are any empty always blocks in the design.

```
//////////////// Example: Document.v /////////////////////////

module test (clock, q_nxt, q);
  input clock, q_nxt;
  output q;
  reg q;
  always @(posedge clock) //warning
  begin
  end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

Violated 1078: empty process is detected.

Rule No 1079 Non-constant Delay

Configurable Parameter

Argument type: none;

Description

The rule checks whether there are any non-constant delay values.

```
//////////////// Example : Document.v //////////////////////////

module test (clock, q_nxt, q);
  input clock, q_nxt;
  output q;
  reg q;
  reg [2:0] a;
  initial
    a = 2;
  always
    #4 a = ~a;
  always @(posedge clock)
    q = #a q_nxt; //warning on "a",
                  //delay value is not static constant
endmodule
```

Run Command :

iverilog-0.8 -tlint Document.v

Linting Tool reports:

Violated 1079: delay value #a is not a constant.

Rule No 1080 Event Not Synthesizable

Configurable Parameter

Argument type: none;

Description

Check to see if there is any event used because it cannot be synthesized.

```
/////////////// Example : Document.v ////////////////////////

module test (clk, rst, d, q);
  input clk, rst, d;
  output q;
  reg q;
  event event1,event2;
  always @(posedge clk)
    if ( !rst )
      ->event1;//"->enent1" non-synthesizable, warning
    else
      ->event2;//"->enent2" non-synthesizable, warning
  always @event1
    q <= 0;
  always @event2
    q <= d;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1080: event "event1" should is not synthesizable.
Violated 1080: event "event2" should is not synthesizable.
```

Rule No 1081 Asynchronous Reference in Edge-sensitive Logic

Configurable Parameter

Argument type: none;

Description

Check to see if there is any signal being read asynchronously in an edge-sensitive logic.

```
/////////////// Example : Document.v ////////////////////////

module test (a, b, clk, reset, chk);
  output a;
  input b, clk, reset, chk;
  reg a;
  always @(posedge clk or posedge reset or posedge chk)
    if (reset)
      a <= 0;
    else if (chk)
      a <= b; //warning here
    else
      a <= a;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1081: signal "b" is being read asynchronously. It may cause simulation-synthesis mismatch.
```

Rule No 1082 Event Enable Statement Not Synthesizable

Configurable Parameter

Argument type: none;

Description

Check to see if there is any event enable statement because it cannot be synthesized.

```
/////////////// Example : Document.v ////////////////////////

module test (c,a,b);
  output c;
  input a,b;
  reg c;
  reg [8*14:1] strvar;
  always @(a or b)
    begin
    if (a == b)
      begin
      c = 'b1;
      strvar = "are equal."; //"are equal." non-synthesizable, warning
      end
    else
      begin
      c = 'b0;
      strvar = "are not equal."; //"are not equal."
                               //non-synthesizable, warning
      end
    end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1082: string "are not equal." should not be used because it is not synthesizable.
```

Rule No 1083 Event Enable Statement Not Synthesizable

Configurable Parameter

Argument type: none;

Description

Check to see if there is any event enable statement because it cannot be synthesized.

```
/////////////// Example : Document.v ////////////////////////

module test (clk, rst, d, q);
  input clk, rst, d;
  output q;
  reg q;
  event event1,event2;
  always @(posedge clk)
    if ( !rst )
      ->event1;//"->enent1" non-synthesizable, warning
    else
      ->event2;//"->enent2" non-synthesizable, warning
  always @event1
    q <= 0;
  always @event2
    q <= d;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1083: event enable statement ->event1 should not be used because it is not synthesizable.
Violated 1083: event enable statement ->event2 should not be used because it is not synthesizable.
```

Rule No 1084 Specify Constant Bit Width Explicitly

Configurable Parameter

Argument type: none;

Description

The rule checks whether there are any constants greater than or equal to 5-bits.

```
//////////////// Example : Document.v /////////////////////////

module smp(A,B,C);
 input[6:0] A,B;
 output[6:0] C;
 reg[6:0] C;
 always@(A or B or C)
  case( A )
   7'd0 : C = B;
   7'd1 : C = 'b000_0010;
   7'd2 : C = 'b000_0100;
   7'd3 : C = 'b000_1000;
   7'd4 : C = 'b001_0000;
   default : C = 'b111_1111;
  endcase
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1084: Specify bit width "7" explicitly when assigning C<0> constant of 5 bits or more.
```

Rule No 1085 One Statement in a Single Always Block

Configurable Parameter

Argument type: none;

Description

This rule check whether there is more than one statement (if/case/while/for/forever/repeat) within a single always block.

```
/////////////// Example : Document.v ////////////////////////

module smp(A,B,C,D,S,W);
 input    A,B,C,D,S;
 output   W;
 reg W;
  always@( A or B or C or D or S )
   begin
     if( A )
       begin
         W = B;
       end
     case( S )
         1'b0 : W = C;
         1'b1 : W = D;
     endcase
   end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

Violated 1085: more than one statement (if/case/while/for/forever/repeat) is described in a single always block.

Rule No 1086 Negative Value Assigned to an Integer

Configurable Parameter

Argument type: none;

Description

This rule checks whether there are any negative values assigned to integers.

```
/////////////// Example : Document.v ////////////////////////

module smp(INA,F);
 input[7:0]   INA;
 output       F;
 reg F;
 integer I;
  always  @(INA) begin : loop
      I=  -1;
      for (I = 0; I <= 7; I = I + 1)
        F = INA[I];
  end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1086: A negative value is assigned to an integer.

Rule No 1087 Missing Process Label Name

Configurable Parameter

Argument type: none;

Description

Check to see if the process block is given a name explicitly.

```
/////////////// Example : Document.v ////////////////////////

module test (a, inc_dec, sum);
  input a, inc_dec;
  output [7:0] sum;
  reg [7:0] sum;
  always @(a or inc_dec)
    begin//: COMBINATIONAL_PROC
    if ( inc_dec == 0)
      sum = a ;
    else
      sum = a ;
    end //good coding style using separate line for each HDL statement
endmodule
```

Run Command :

iverilog-0.8 -tlint Document.v

Linting Tool reports:

Violated 1087: process should be named.

Rule No 1088 Multiple Top Modules

Configurable Parameter

Argument type: none;

Description

Check to see if there is more than one top module in the design.

```
/////////////// Example : Document.v ////////////////////////

module initval;
  reg clock, reset;
  wire [8:0] count;
  initial
    begin
    clock=0;
    reset=0;
    $monitor($time,,,"count=%d,reset=%d",count,reset);
    #200 $finish;
    end
  always
    #4 clock=!clock;
  //test cc (clock, reset, count);
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1088: more than one top module detected, top modules.

```
//////////////// Example : Document.v /////////////////////////

module test (clock, reset, count); //warning on 'test',
                                  //another top module
  input clock, reset;
  output [8:0] count;
  reg [8:0] count;
  initial
    count <= 0;
  always @(posedge clock or posedge reset)
    begin
    if (reset)
      count = 0;
    else
      count= count + 1;
    end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1088: more than one top module detected, top modules.
```

Rule No 1089 Initial Block Not Synthesizable

Configurable Parameter

Argument type: none;

Description

Check to see if there is any initial block because it cannot be synthesized.

```
//////////////// Example : Document.v //////////////////////////

module test;
  reg a, b;
  initial //non-synthesizable, warning
  begin
    a = 0;
    b = 1;
  end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1089: Initial Block Not Synthesizable.
```

Rule No 1090 Tri-state Enable Mixed with Other

Configurable Parameter

Argument type: none;

Description

This rules checks whether the driving logics for the enable condition of a tri-state buffer is multi input.

```
//////////////// Example : Document.v //////////////////////////

module test(tri_out1,data,a,b);
  input data,a,b;
  output tri_out1;
  wire tri_out1, rden, data;
  assign tri_out1 = ( rden ) ? data : 1'bz;
  and AA(rden,a,b); //warning here
  //not AA(rden,b);
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1090: Tri-state Enable "rden" Mixed with other logic in module.
```

Rule No 1091 Duplicate Signal Found in Sensitivity List

Configurable Parameter

Argument type: none;

Description

Check to see if there is any duplicate signal used in sensitivity list.

```
//////////////// Example : Document.v //////////////////////////


module test (count, clk, rst_p);
  input clk, rst_p;
  output [7:0] count;
  reg [7:0] count;

  always @(posedge clk or clk  //warning on "clk"
            or posedge rst_p)
    if (rst_p)
      count = 0;
    else
      count = 1;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1091: Edge Signal "clk" and Non Edge Signal "clk" are duplicate.
```

Rule No 1092 Tri-state Output Mixed with Other Logic in a Module

Configurable Parameter

Argument type: none;

Description

This rules checks whether the fan-out logic of a tri-state buffer is multi input.

```
//////////////// Example : Document.v //////////////////////////

module smp1(rden, wren, data, tri_out1, cnt1);

  input rden, wren, data;
  output tri_out1;
  output cnt1;

   assign tri_out1 = ( rden ) ? data : 1'bz;
   assign cnt1 = tri_out1 & wren;    //warning here

 endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1092: Tri-state Output "tri_out1" Mixed with Other Logic in a Module.

Rule No 1093 Constant Event Expression

Configurable Parameter

Argument type: none;

Description

The rule checks whether there is a constant used as an event control.

```
//////////////// Example : Document.v //////////////////////////

module test (a, y);
  input [7:0] a;
  output [7:0] y;
  reg [7:0] y;
  parameter c = 1;
  always @(c) //warning
    y = a;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1093: Constant Event Expression "c<0>".
```

Rule No 1094 Clock Driven by Sequential Logic

Configurable Parameter

Argument type: none;

Description

Check to see if there is any clock signal driven by a sequential logic.

```
/////////////// Example : Document.v ////////////////////////

module test (count, reset, clock, data);
  input clock, reset, data;
  output [8:0] count;
  reg [8:0] count;
  reg qc;
  always @(posedge clock)
    qc = data; //"qc" is output of register, warning on "qc"
  always @(posedge qc or negedge reset)
    begin //qc is used as clock signal and drived
          //by above sequential logic
    if (~reset)
      count = 0;
    else
      count= count + 1;
    end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1094: Clock Signal "qc" Used as Sequential Logic.

Rule No 1095 Clock Signal Used as Data Input

Configurable Parameter

Argument type: none;

Description

The rule checks whether there are any clock signals used as a data input of any storage elements.

```
/////////////// Example : Document.v ////////////////////////

module test (clock2, clock1, data, y1, y2);
  input clock2, clock1, data;
  output y1, y2;
  reg y1, y2;
  always @(posedge clock1)
    y1 = clock2; //warning
  always @(posedge clock2)
    y2 = data;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1095: Clock Signal "clock2" Used as Data Input.
```

Rule No 1096 Logic Expression Used in Sensitivity List

Configurable Parameter

Argument type: none;

Description

The rule checks whether there are any logic expressions used in any sensitivity list.

```
/////////////// Example : Document.v ////////////////////////

module test (count, clk1, clk2, rst_p);
  input clk1, clk2, rst_p;
  output [7:0] count;
  reg [7:0] count;
  always @(posedge(clk1|clk2) or //warning
           posedge (rst_p))
    if (rst_p)
      count = 0;
    else
      count = count + 1;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1096: Logic Expression Used in Sensitivity List.

Rule No 1097 Direct Connection from Input to Output

Configurable Parameter

Argument type: none;

Description

Check to see if there is any input signal that is connected directly to an output signal.

```
/////////////// Example: Document.v ////////////////////////

module test (a, c);
  input a; //warning on 'a', is directly connected to output signal 'c'
  output c;
  reg c;
  always @(a)
    c = a;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1097: violation as Direct Connection from Input "a" to Output "c".
```

Rule No 1098 Bi-directional Port Declared

Configurable Parameter

Argument type: none;

Description

Checking for bi-directional ports.

```
/////////////// Example : Document.v ////////////////////////

module test (a, b, c, d, f, f1, f2);
  input a, b, c, d;
  output f;
  inout f1, f2; //warning on "f1", "f2"
  and AA(f,f1,f2);
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1098: Bidirectional Port "f1" Decleared.
Violated 1098: Bidirectional Port "f2" Decleared.
```

Rule No 1099 Signal Driven by Constant

Configurable Parameter

Argument type: none;

Description

The rule checks whether there is a signal driven by a constant.

```
/////////////// Example : Document.v ////////////////////////

module test;
  parameter W = 2;
  wire  q;
  assign q = 2 * W; //warning
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1099: Signal "q" Driven By Constant.
```

Rule No 1100 Assignment to an Input Signal

Configurable Parameter

Argument type: none;

Description

The rule checks whether there are any input signals assigned with a value.

```
/////////////// Example : Document.v ////////////////////////

module test (a, b, c);
  input [7:0] a, b;
  output [7:0] c;
  reg [7:0] c;
  wire [7:0] a;
  assign a[2] = 0; //warning on "a[2]", input signal is assigned
  always @(a)
    c = b;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1100: Assignment to a Signal.

Rule No 1101 Reset Signal Active High and Low

Configurable Parameter

Argument type: none;

Description

Check to see if there is any reset signal that is both active high and active low.

```
/////////////// Example : Document.v ////////////////////////

module test (clock1, clock2, reset, count, data, y);
  input clock1,clock2,reset,data;
  output [8:0] count;
  reg [8:0] count;
  output y;
  reg y;
  initial
    count <= 0;
  always @(posedge clock1 or negedge reset)
    begin
    if (~reset) //low active "reset", warning on "reset"
      count = 0;
    else
      count = count + 1;
    end
  always @(posedge clock2)
    if (reset) //high active "reset"
      y = 0;
    else
      y = data;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1101: Reset Signal "reset" Active High and Low.
```

Rule No 1102 Multiple Clock Signals

Configurable Parameter

Argument type: none;

Description

Check to see if there is more than one clock signal in a module.

```
//////////////// Example : Document.v //////////////////////////

module test (clock1,clock2,reset,count,data,y);
  input clock1, clock2, reset, data;
  output [8:0] count;
  reg [8:0] count;
  output y;
  reg y;
  always @(posedge clock1 or negedge reset)//clock signal "clock1"
    begin
    if (~reset)
      count = 0;
    else
      count= count + 1;
    end
  always @(posedge clock2)//warning on "clock2", another clock signal
    begin
      if (~reset)
         y = 0;
      else
         y = data;
    end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1102: more than one clock signal in a module.
```

Rule No 1103 Signal Stuck at Logic 0

Configurable Parameter

Argument type: none;

Description

The rule checks whether there is a signal whose value is always logic 0.

```
//////////////// Example : Document.v //////////////////////////

module test (d, e, a, b);
  input a, b;
  output d,e;
  reg d,e;
  always @( a )
    begin
      d = 0; //warning
      e = a & ~a; //warning
    end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1103: Signal "d" Stuck at Logic 0.
Violated 1103: Signal "a" Stuck at Logic 0.
```

Rule No 1104 Signal with No Load

Configurable Parameter

Argument type: none;

Description

The rule checks whether there are any signals without a load.

```
//////////////// Example : Document.v //////////////////////////

module test (c, a, b);
  input [1:0] a, b;
  output [1:0] c;
  reg [1:0] c;
  wire e; //warning
  wire d; //warning
  assign e = 1;
  assign d = 0;
  always @(a or b)
    c = a + b;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1104: Signal "d" with no Load.
```

Rule No 1105 Integer Used in Concatenation

Configurable Parameter

Argument type: none;

Description

The rule checks for integers used in a concatenation.

```
//////////////// Example : Document.v //////////////////////////

module test (out, in);
  output [7:0] out;
  input in;
  wire [7:0] out;
  integer i;
  assign out = {in, i}; //warning
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1105: Integer "test.i" is used in concatenation.
```

Rule No 1106 Variable Name Case

Configurable Parameter

Argument type: none;

Description

Check to see if the variable names, including integer, real and realtime variables, are all in lower (or upper) case.

```
/////////////// Example : Document.v ////////////////////////

module test (a, b, y);
  input [7:0] a, b;
  output [7:0] y;
  reg [7:0] y;
  integer N; //warning on 'N' if CASE_LOWER
  always @(a)
    begin
    y=0;
    for (N=0; N<=7; N=N+1)
      y[N] = a[N] & b[N];
    end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1106: Integer variable "N" should be named in CASE_LOWER case.
```

Rule No 1107 Variable Name Too Long

Configurable Parameter

Argument type: none;

Description

Check to see if the length of the variable name exceeds 'length' characters.

```
/////////////// Example : Document.v ////////////////////////

module test (a, b,  y);
  input [7:0] a, b;
  output [7:0] y;
  reg [7:0] y;
  integer Nnnnnnnnnnnn; //warning on 'N' if CASE_LOWER
  always @(a)
    begin
    y=0;
    for (Nnnnnnnnnnnn=0; Nnnnnnnnnnnn<=7; Nnnnnnnnnnnn=Nnnnnnnnnnnn+1)
      y[Nnnnnnnnnnnn] = a[Nnnnnnnnnnnn] & b[Nnnnnnnnnnnn];
    end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1107: The length of Integer variable name "Nnnnnnnnnnnn" should not exceed 10 characters.
```

Rule No 1108 Variable Name Prefix or Suffix

Configurable Parameter

Argument type: none;

Description

Check to see if the variable name has a recommended prefix or suffix.

```
//////////////// Example : Document.v //////////////////////////

module test (a, b,  y);
  input [7:0] a, b;
  output [7:0] y;
  reg [7:0] y;
  integer Nnnnnnnnnnnn; //warning
  always @(a)
    begin
    y=0;
    for (Nnnnnnnnnnnn=0; Nnnnnnnnnnnn<=7; Nnnnnnnnnnnn=Nnnnnnnnnnnn+1)
      y[Nnnnnnnnnnnn] = a[Nnnnnnnnnnnn] & b[Nnnnnnnnnnnn];
    end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1108: variable name "Nnnnnnnnnnnn" does not match to regular expression v_.*.
```

Rule No 1109 Signal Stuck at Logic 1

Configurable Parameter

Argument type: none;

Description

The rule checks whether there is a signal whose value is always logic 1.

```
/////////////// Example : Document.v ////////////////////////

module test (d, e, a, b);
  input a, b;
  output d, e;
  reg d, e;
  wire a, b;
  always @( a )
    begin
      d = 1; //warning
      e = a | ~a; //warning
    end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1109: Signal "a" Stuck at Logic 1.
```

Rule No 1110 Set Signal Used as Data Input

Configurable Parameter

Argument type: none;

Description

Check to see if there is any set signal that is also used as a data signal.

```
//////////////// Example : Document.v /////////////////////////

module test (clock2, set, data, y1, y2);
  input clock2, data, set;
  output y1, y2;
  reg y1, y2;
  always  @(posedge clock2 or negedge set )
    if (~ set )//"set" is used as set signal
      y1  <= 1'b1;
    else
      y1  <= data;
   always @(posedge clock2)
    y2 = set;//"set" is used as data input, warning on "set"
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1110: Set Signal "set" Used as Data Input.
```

Rule No 1111 Signal Used as Set and Reset

Configurable Parameter

Argument type: none;

Description

Check to see if there is any reset signal that is also used as a set signal.

```
//////////////// Example : Document.v //////////////////////////

module test (q1, q2, clk_1, enable, d1, d2);
  output q1, q2;
  input clk_1, enable, d1, d2;
  reg q1, q2;
  wire clk_1, enable, d1, d2;
  always  @( posedge clk_1 or negedge enable )
    if (  ~enable ) //"enable" is used as reset
      q1  <= 1'b0;
    else
      q1  <= d1;
  always @( posedge clk_1 or posedge enable )
    if ( enable ) //warning here, "enable" is also used as
                  //set signal
      q2 <= 1'b1;
    else
      q2 <= d2;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1111: "enable" signal is used as both set and reset.

Rule No 1112 Tri-state Inferred in Non-top Module

Configurable Parameter

Argument type: none;

Description

Check to see there is any tri-state inferred in a non-top module.

```
//////////////// Example : Document.v /////////////////////////

module top(count, clock, reset, load, data);
  output  count;
  input  data;
  input clock, reset, load;
  wire  data;
  wire clock, reset, load;
  wire  count;
  up_counter u_up_counter_1(count, clock, reset, load, data);
endmodule
module up_counter(out, clock, reset, load, data);
  output  out;
  input  data;
  input clock, reset, load;
  wire  data;
  wire clock, reset, load;
  wire  out;
  reg  count;
  assign out = load ? 1'bz : count;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint –S Document.v
```

Linting Tool reports:

```
Violated 1112: Nontop Module Tri-state "out" Detected.
```

Rule No 1113 Combinational Path Between Two Registers is Too Long

Configurable Parameter

Argument type: none;

Description

Heck to see if there is any combinational path, that is too long, between two registers.

```
/////////////// Example : Document.v ////////////////////////

module test (clk, a, tri_, R);
  input clk, a, tri_;
  output R;
  wire clk, a;
  reg R, R1;
  wire c, d, e;
  always @(posedge clk)
    R1 = a;
  assign c = R1;
  assign d = tri_ ? c : 1'bz;
  always @(posedge clk)
    R = d;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1113: Combinational Path Too Long.
```

Rule No 1114 Bit of a Bus Signal Used as Special Type Signal

Configurable Parameter

Argument type: none;

Description

Check to see if any bit select used as some special type signal.

```
/////////////// Example : Document.v ////////////////////////

module test (q, clk, rst, in);
  input in, clk;
  input [1:0] rst;
  output q;
  reg q;
  always @(posedge clk)
    if (~rst[1]) //reset signal "rst[1]" is a bit select
      q <= 0;
    else
      q <= in;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1114: Special signal "rst<1>" is a bit select.
```

Rule No 1115 Real Variable Not Synthesizable

Configurable Parameter

Argument type: none;

Description

Check to see if there is any real variable used because it cannot be synthesized.

//////////////// Example: Document.v //////////////////////////

```
module test;
  real real_b;//non-synthesizable, warning
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1115: Variable "real" "real_b" Not Synthesizable.
```

//////////////// Example : Document.v //////////////////////////

```
module test;
  realtime rt;//non-synthesizable, warning
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1115: Variable "real" "rt" Not Synthesizable.
```

Rule No 1116 Memory Not Synthesizable

Configurable Parameter

Argument type: none;

Description

Check to see if there is any memory

```
/////////////// Example : Document.v ////////////////////////

module test(c,a,b);
  reg [7:0] mem[0:4096]; //warning here, memory length bits too large
  input [1:0] a,b;
  output [1:0] c;
  reg [1:0] c;
  always @( a or b )
    c = {a[1], b[0]};
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1116: Memory "mem" Not Synthesizable.
```

Rule No 1117 Too Many Words in Memory

Configurable Parameter

Argument type: none;

Description

Check to see if the memory words length exceeds the 'length' words.

```
/////////////// Example : Document.v ////////////////////////

module test(c,a,b);
  reg [7:0] mem[0:4096]; //warning here, memory length bits too large
  input [1:0] a,b;
  output [1:0] c;
  reg [1:0] c;
  always @( a or b )
    c = {a[1], b[0]};
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1117: Too Many Words "4097" in Memory "mem".
```

Rule No 1118 Too Bits Many in Memory

Configurable Parameter

Argument type: none;

Description

Check to see if the memory length exceeds the 'length' bits.

```
/////////////// Example : Document.v ////////////////////////

module test(c,a,b);
  reg [7:0] mem[0:4096]; //warning here, memory length bits too large
  input [1:0] a,b;
  output [1:0] c;
  reg [1:0] c;
  always @( a or b )
    c = {a[1], b[0]};
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1118: Too Many Bits "32776" in Memory.
```

Rule No 1119 Net Types Not Synthesizable

Configurable Parameter

Argument type: none;

Description

Check to see if there is any net of type tri1, supply0, triand, tri0, supply1, trior, or trireg,because it cannot be synthesized.

```
/////////////// Example : Document.v ////////////////////////

module test(a,b,t);
input a,b;
output t;
//tri0 t;
 tri0 t;
bufif0 B(t,a,b);
/*

  tri1 tri1_t;//"tri1" non-synthesizable, warning
  tri0 tri0_t;//"tri0" non-synthesizable, warning
  supply0 supply0_t;//"supply0" non-synthesizable, warning
  supply1 supply1_t;//"supply1" non-synthesizable, warning
  triand triand_t;//"triand" non-synthesizable, warning
  trior trior_t;//"trior" non-synthesizable, warning*/
//trireg trireg_t;//"trireg" non-synthesizable, warning
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1119: Net Type "tri0" "t" Not Synthesizable.
```

Rule No 1120 Both Edge and Non-edge Expressions in the Sensitivity List

Configurable Parameter

Argument type: none;

Description

Check edge non edge.

```
/////////////// Example : Document.v ////////////////////////

module test (clk, rst, d, q);
  input clk, rst;
  input d;
  output q;
  reg q;
  always @( posedge clk or rst ) //warning
    if( rst )
      q <= 1'b0;
    else
      q <= d;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1120: Both Edge "clk" and Non-edge "rst" Expressions in the Sensitivity List.

Rule No 1121 Non-constant Bit Range

Configurable Parameter

Argument type: none;

Description

Check to see if non-constant is used as an index or range of a signal because it cannot be synthesized.

```
/////////////// Example : Document.v ////////////////////////

module test (out, s, in);
  parameter S_WIDTH = 3;
  parameter DATA_WIDTH = 8;
  output out;
  reg out;
  input [S_WIDTH-1:0] s;
  input [DATA_WIDTH-1:0] in;
  always @(in)
    out = in[s]; //"s" is not a constant in bits selection, warning
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1121: "s<2>" not a constant in bits selection.
```

Rule No 1122 Multiple Bits in Special Type Signal

Configurable Parameter

Argument type: none;

Description

Check to see if any bus used as some special type signal.

```
/////////////// Example : Document.v ////////////////////////

module test (q, clk, rst, in);
  input in, clk;
  input [1:0] rst;
  output q;
  reg q;
  always @(posedge clk)
    if (rst) //reset signal "rst" is multiple bit signal
      q <= 0;
    else
      q <= in;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1122: Multiple Bits in Special Type "rst<0>" Reset Signal.
```

Rule No 1123 UDP Not Synthesizable

Configurable Parameter

Argument type: none;

Description

Check to see if there is any UDP declaration because it cannot be synthesized.

```
/////////////// Example : Document.v ////////////////////////

module test_M(a,b,c,d);
 input a,b,c;
 output d;
test test1 (d,a,b,c);
endmodule
primitive test(sum, cin, a, b);//non-synthesizable, warning
  output sum;
  input cin, a, b;
  table
  0 0 0 : 0;
   0 0 1 : 1;
  0 1 0 : 1;
  0 1 1 : 0;
  1 0 0 : 1;
  1 0 1 : 0;
  1 1 0 : 0;
  1 1 1 : 1;
  endtable
endprimitive
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1123: UDP declaration "test" should not be used because it is not synthesizable.

Rule No 1124 a Combinational Path Between PI and PO without Being Registered

Configurable Parameter

Argument type: none;

Description

Check to see if there is a combinational path between PI and PO without being registered.

```
/////////////// Example : Document.v ////////////////////////

module test (a, b, c, k);
  input a;
  input b;
  output c;
  input [2:0] k;
  wire c;
  assign c = a & b & k[0];
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1124: Combinational Path Between PI "k" and PO "c" without Being Registered.
Violated 1124: Combinational Path Between PI "a" and PO "c" without Being Registered.
Violated 1124: Combinational Path Between PI "b" and PO "c" without Being Registered.
```

Rule No 1125 Non-constant Divisor

Configurable Parameter

Argument type: none;

Description

Check to see if there is any division or modulo operation taking a non-constant as its second operand.

```
//////////////// Example : Document.v //////////////////////////

module test (a, b, y);
  input [2:0] a, b;
  output [3:0] y;
  reg [3:0] y;
  always @(a or b)
    y = a / b ;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1125: Non-constant Divisor "b<2>".
```

Rule No 1126 Non-constant Dividend

Configurable Parameter

Argument type: none;

Description

Check to see if there is any division or modulo operation taking a non-constant as its first operand.

```
/////////////// Example : Document.v ////////////////////////

module test (a, b, y);
  input [2:0] a, b;
  output [3:0] y;
  reg [3:0] y;
  always @(a or b)
    y = a / b ;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1126: Non-constant Divident "a<2>".
```

Rule No 1127 Signal Name Prefix or Suffix

Configurable Parameter

Argument type: none;

Description

Check to see if the signal name has a recommended prefix or suffix.

```
/////////////// Example : Document.v ////////////////////////

module test(a,b);
input a;
output b;
not N(b,a);
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1127: signal name "a" does not match to regular expression s_.
Violated 1127: signal name "b" does not match to regular expression s_.
```

Rule No 1128 Sign/Unsigned Conversion in Assignment

Configurable Parameter

Argument type: none;

Description

Check signed/unsigned conversion in assignment..

```
//////////////// Example : Document.v /////////////////////////

module test;
  wire [7:0] regA;
  reg signed [7:0] regS;
  always @( regA )
    regS = regA; //warning here
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1128: unsigned "regA" to signed "regS" assignment occurs.
```

Rule No 1129 Function with Integer Return Value

Configurable Parameter

Argument type: none;

Description

The rule checks whether there are any functions whose return value is an integer.

```
/////////////// Example : Document.v ////////////////////////

module test;
  function integer TF; //warning
    input a, b, cin;
    TF = a + b + cin;
  endfunction
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1129: Integer function "TF" detected.
```

Rule No 1130 Function Name Length

Configurable Parameter

Argument type: none;

Description

Check to see if the length of the function name is in the specified range

```
/////////////// Example : Document.v ////////////////////////

module test;
  function [12:0] mu; //warning on "mu"
  input [12:0] a;
  input [15:0] b;
  begin: serialMult
  reg [5:0] mcnd,mpy;
    mpy = b[5:0];
    mcnd = a[5:0];
    mu=0;
    repeat(6) begin
      if(mpy[0])
        mu = mu + {mcnd,6'b000000};
      mu = mu >> 1;
      mpy = mpy >> 1;
    end
  end
  endfunction
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1130: The length of function name "mu" is unconventional and should be in the range from 3 to 16.

Rule No 1131 Function Name Case

Configurable Parameter

Argument type: none;

Description

Check to see if the function names are all in lower (or upper) case.

```
/////////////// Example : Document.v ////////////////////////

module test;
  function Func_add;
           //warning on "Func_add"
  input a;
  input b;
  begin
  end
  endfunction
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1131: function "Func_add" should be named in CASE_LOWER case.
```

Rule No 1132 Function Name Prefix or Suffix

Configurable Parameter

Argument type: none;

Description

Check to see if the function name has a recommended prefix or suffix.

```
/////////////// Example : Document.v ////////////////////////

module test;
  function add; //warning on "add", using "f_add" like
  input a;
  input b;
  begin
  end
  endfunction
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1132: function name "add" does not match to regular expression f_.
```

Rule No 1133 Task Name Length

Configurable Parameter

Argument type: none;

Description

Check to see if the length of the task name is in the specified range

```
//////////////// Example : Document.v //////////////////////////

module test;
  task Check_0123456789012; //warning on "Check_0123456789012"
  begin
  end
  endtask
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1133: task "Check_0123456789012" should be named in CASE_LOWER case.
```

Rule No 1134 Task Name Case

Configurable Parameter

Argument type: none;

Description

Check to see if the task names are all in lower (or upper) case.

```
/////////////// Example : Document.v ////////////////////////

module test;
  task Check; //warning on "Check"
  begin
  end
  endtask
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1134: task "Check" should be named in CASE_LOWER case.
```

Rule No 1135 Task Name Prefix or Suffix

Configurable Parameter

Argument type: none;

Description

Check to see if the task name has a recommended prefix or suffix.

```
/////////////// Example : Document.v /////////////////////////

module test;
  task Check; //warning on "Check", using "p_Check" like
  begin
  end
  endtask
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1135: Task name "Check" does not match to regular expression p_.
```

Rule No 1136 Module Name Length

Configurable Parameter

Argument type: none;

Description

Check to see if the length of the module name is in the specified range

```
/////////////// Example : Document.v ////////////////////////

module test_top_0123456789012;
//warning on "test_top_0123456789012"
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1136: Module Name "test_top_0123456789012" is unconventional and should be in the range from 3 to 16.

Rule No 1137 Module Name Case

Configurable Parameter

Argument type: none;

Description

Check to see if the module names are all in lower (or upper) case.

```
//////////////// Example : Document.v //////////////////////////

module TEST_TOP; //warning on "TEST_TOP"
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1137: module "TEST_TOP" should be named in CASE_LOWER case.
```

Rule No 1138 Module Name Prefix or Suffix

Configurable Parameter

Argument type: none;

Description

Check to see if the module names has a recommended prefix or suffix.

```
/////////////// Example : Document.v ////////////////////////

module TEST; //warning on "TEST_TOP", using
             //"TOP_module" like
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1138: module name "TEST" does not match to regular expression .*_module.
```

Rule No 1139 Combinational Logic Found in Sequential Block

Configurable Parameter

Argument type: none;

Description

Check to see if any combinational expression or statement mixed in sequential block.

```
/////////////// Example : Document.v ////////////////////////

module test (clock, reset, a, b, c, d, in1, in2, out);
  input clock;
  input reset;
  input a, b, c, d, in1, in2;
  output out;
  reg     out;
  always @(posedge clock) begin
    if ( ( a & b )== 0 ) // warning here
      out = 0;
    else if ( c & d ) // warning here
      out = 1;
    else
      out = ( in1 & in2 ); // warning here
  end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1139: Combinational Logic Found in Sequential "clock" Block.
```

Rule No 1140 unsigned Vector Compared with a Negative Value

Configurable Parameter

Argument type: none;

Description

Check to see if there is any unsigned vector compared with a negative value.

```
//////////////// Example : Document.v //////////////////////////

module test (out0,in1,in2,sel);
  input sel;
  input [1:0] in1,in2;
  output [1:0] out0;
  reg [1:0] out0;
  always @( in1 or in2 or sel)
    if ( sel == 1'b1 )
      out0 = (in1 > -2'd1); //warning here
    else
      out0 = in2;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1140: Unsigned Vector Compared with a Negative Value "111".
```

Rule No 1141 Logical Operators Used in Single-bit Operations

Configurable Parameter

Argument type: none;

Description

This rule checks whether single-bit operations use bitwise operations

```
//////////////// Example : Document.v //////////////////////////

module smp(out, in1, in2, in3);
 output out;
 input in1, in2, in3;
  assign out = !(in1 & in2 | in3);
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1141: bitwise operators should be used instead of logic "NotOp" operators in single-bit operations.

Rule No 1142 Conditional Assignment Detected

Configurable Parameter

Argument type: none;

Description

Check to see if there is any conditional assignment used,

```
/////////////// Example : Document.v ////////////////////////

module test (out0,in1,in2,sel);
  input sel;
  input  in1,in2;
  output  out0;
  reg  out0;
  always @( in1 or in2 or sel)
    out0 = sel ? in1 : in2; //warning here
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

Violated 1142: no conditional assignment allowd on "sel".

Rule No 1143 'reg' Declaration Detected

Configurable Parameter

Argument type: none;

Description

The rule checks whether there are any 'reg' declarations in the design.

```
/////////////// Example : Document.v ////////////////////////

module test (clk,count);
  input clk;
  output [8:0] count;
  reg [8:0] count;
  reg [8:0] nextCount;
  always @(posedge clk)
      count = nextCount;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1143: 'reg' declaration "nextCount" is detected.
```

Rule No 1144 Wire Declaration Detected

Configurable Parameter

Argument type: none;

Description

The rule checks whether there are any 'wire' declarations in the design.

```
/////////////// Example : Document.v ////////////////////////

module test;
  wire a,b;
  not A(a,b);
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1144: 'wire' declaration "a" is detected.
Violated 1144: 'wire' declaration "b" is detected.
```

Rule No 1145 Multiple Clock Source Not Recommended

Configurable Parameter

Argument type: none;

Description

Check if the design use only one clock source.

```
/////////////// Example: Document.v ////////////////////////

module test;
  wire clk2, clk1, d, set; //2 clk sources
  reg q;
  always @(posedge clk2) begin
    if (set) q <= 1'b1;
    else q<=d;
  end
  always @(posedge clk1) begin
    if (set) q<= 1'b1;
    else q<=d;
  end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1145: Multiple Clock Source Not Recommended.

Rule No 1146 Clock Signal Used as Reset

Configurable Parameter

Argument type: none;

Description

Violated 1146: Clock Signal "clock" Used as Reset.

```
//////////////// Example : Document.v //////////////////////////

module test (clock, clock2, reset, count, d);
  input clock, clock2, reset, d;
  output [8:0] count;
  reg [8:0] count;
  reg a;
  initial
    count<=0;
  always @(posedge clock or negedge reset)  //"clock" is used as clock
    begin
    if (reset)
      count = 0;
    else
      count= count + 1;
    end
  always @(posedge clock2)
    if (clock) //warning
      a=0;
    else
      a=d;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1146: Clock Signal "clock" Used as Reset.

Rule No 1147 Current State Name Prefix or Suffix

Configurable Parameter

Argument type: none;

Description

Check to see if the current state register names have recommended prefix or suffix.

```
//////////////// Example : Document.v //////////////////////////

module fsm(out, in, clock, reset);
output out;
input in, clock, reset;
reg out;
reg [1:0] current, next;
always @(in or current) begin
  out = ~current[1] & current[0];
  next = 0;
  if (current == 0)
    if (in)
       next = 1;
  if (current == 1)
    if (in)
       next = 3;
  if (current == 3)
    if (in)
       next = 3;
  else
    next = 1;
end
always @(posedge clock or negedge reset) begin
  if (~reset)
    current <= 0;
  else
    current <= next; //warning here; good style if using "_cs"
                     //as suffix of current state register "current"
  end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1147: State register name "current<0>" does not match to regular expression .*_cs.
Violated 1147: State register name "current<1>" does not match to regular expression .*_cs.
```

Rule No 1148 Set Driven by Sequential Logic

Configurable Parameter

Argument type: none;

Description

Check to see if there is any set signal driven by sequential logic.

```
/////////////// Example : Document.v ////////////////////////

module test (q, clk, c, d);
  output q;
  output [3:0] c;
  input clk, d;
  reg q;
  reg [3:0] c;
  always @(posedge clk)
    c <= c + 1; //warning on "c[3]", set
                  //signal is an output of
                  //sequential device
  always @(posedge clk or posedge c[3])
    if (~c[3])
      q <= 1;
    else
      q <= d;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1148: Set "c<3>" Driven by Sequential Logic.
```

Rule No 1149 No Set or Reset Signal

Configurable Parameter

Argument type: none;

Description

Check to see if there is any register without reset/set control.

```
//////////////// Example : Document.v //////////////////////////

module test (counter, clock, reset);
  input clock, reset;
  output [3:0] counter;
  reg [3:0] counter;
  wire [3:0] tmp;
 pure_dff  pure_dff1( tmp, clock, tmp+4'b1 );
  always @( reset or tmp )
    if ( ~reset )
      counter <= 0;
    else
      counter <= tmp;
endmodule
module pure_dff (q, clock, d);
  output [3:0] q;
  input clock;
  input [3:0] d;
  reg [3:0] q;
  always @(posedge clock)
    q = d; //when simulation begins, the value of "q" cannot
           //be easily determined, warning
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1149: No Set or Reset Signal only Clock "clock".

Rule No 1150 Synchronous Loop

Configurable Parameter

Argument type: none;

Description

Check to see if there are any synchronous loops in which all the registers without set/reset.

```
/////////////// Example : Document.v ////////////////////////

module test (count, clk);
  parameter number = 10;
  output [3:0] count;
  input clk;
  reg [3:0] count;
  reg i_rst;
  always @(posedge clk)
    count <= count + 1; //warning here
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1150: Synchronous Loop count<3> --> count<3> without set reset.
```

Rule No 1151 Asynchronous Loop

Configurable Parameter

Argument type: none;

Description

Check to see if there are any asynchronous loops in the design.

```
/////////////// Example : Document.v ////////////////////////

module test (count, clk);
  parameter number = 10;
  output [3:0] count;
  input clk;
  reg [3:0] count;
  reg i_rst;
  always @(posedge clk or posedge i_rst) //warning on
                              //"count->i_rst->count"
    if (i_rst )
      count <= 0;
    else
      count <= count + 1;
  always @( count )
    if ( count == number )
      i_rst = 1;
    else
      i_rst = 0;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1151: Asynchronous Loop count<3> --> i_rst --> count<3>.
```

Rule No 1152 Reset Name Prefix or Suffix

Configurable Parameter

Argument type: none;

Description

Check to see if the reset signal names have recommended prefix or suffix.

```
/////////////// Example : Document.v ////////////////////////

module test (q, clk, en, reset, d);
  output q;
  input clk, en, reset, d;
  reg q;
  wire clk, en, reset, d;
  wire rst_en;
  and U_and_1(rst_en, reset, en);//warning on "rst_en", is gated
  always @( posedge clk or negedge rst_en )
    if (rst_en )
      q <= 1'b0;
    else
      q <= d;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1152: reset signal name "rst_en" does not match to regular expression rst_.
```

Rule No 1153 Set Name Prefix or Suffix

Configurable Parameter

Argument type: none;

Description

Check to see if the set signal names have recommended prefix or suffix.

```
/////////////// Example : Document.v ////////////////////////

module test (q, clk, en, set, d);
  output q;
  input clk, en, set, d;
  reg q;
  wire clk, en, set, d;
  wire set_en;
  parameter P = 2;
  and U_and_1(set_en, set, en); //warning on "set_en", is gated
  always @( posedge clk or negedge set_en )
    if (~set_en )
      q <= 1'b1;
    else
      q <= d;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1153: set signal name "set_en" does not match to regular expression set_.
```

Rule No 1154 Test Clock Passed Only One 2x1 Multiplex

Configurable Parameter

Argument type: none;

Description

Check to see if all F/F in design should get the test-clock from test-clock source with only a 2x1 multiplexer.

```
/////////////// Example : Document.v ////////////////////////

module top(q,d,c1,c2,c3,sel,set,reset);
output q;
input d,c1,c2,c3,sel,set,reset;
reg q;
wire clk;
wire c;
wire c4;
buf buffer(c,c1);
assign clk=sel?c:c4;
assign c4=sel?c2:c3;
always @(posedge clk or posedge set or negedge reset)
        if(~reset)q=1'b0;
        else if(set)q=1'b1;
        else q=d;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1154: Test Clock "clk" Passed Only One 2x1 Multiplex.
```

Rule No 1155 Clock Signal Used as a Control

Configurable Parameter

Argument type: none;

Description

The rule checks whether there are any clock signals used as control signals.

```
/////////////// Example : Document.v ////////////////////////

module test (clock2, data, y1, y2);
  input clock2, data;
  output y1, y2;
  reg y1, y2;
  always @(posedge clock2) //"clock2" is used as clock
    y1 = data;
  always  @( data or clock2 )
    if (  clock2 ) //warning
      y2  = data;
    else
      y2  = ~data;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1155: Clock Signal "clock2" Used as a Control.
```

Rule No 1156 Test Clock should be Resolved to Primary Input

Configurable Parameter

Argument type: none;

Description

Check to see if clock source signal is resolved to PI.

```
//////////////// Example : Document.v //////////////////////////

module top(q,d1,d2,ctrl,clk,reset);
output q;
input d1,d2,ctrl,clk,reset;
wire gen_clk;
reg q;
reg o;
always @(posedge clk or negedge reset)
  if(~reset)o=1'b0;
  else o=d1;
assign gen_clk=o&ctrl;//report here
always @(posedge gen_clk)
   q=d2;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1156: Test Clock "gen_clk" should be Resolved to Primary Input.

Rule No 1157 Clock Feeds into Primary Output

Configurable Parameter

Argument type: none;

Description

The rule checks whether there are any clock signals feeding into primary outputs directly or indirectly.

```
/////////////// Example : Document.v ////////////////////////

module test (clock, reset, count, clk_out);
  input clock, reset;
  output [8:0] count;
  output clk_out;
  reg [8:0] count;
  initial
    count<=0;
  always@(posedge clock or negedge reset)//"clock" is used as clock
    begin
    if (reset)
      count = 0;
    else
      count= count + 1;
    end
  assign clk_out = clock; //warning
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1157: Clock "clock" feeds the Primary Output signal directly or indirectly.
```

Rule No 1158 Test Clock Signal Used as Data Input

Configurable Parameter

Argument type: none;

Description

Check to see if test clock be used as data input of register.

```
/////////////// Example : Document.v ////////////////////////

 module top(q1,q2,d,clk1,clk2,reset);
  output q1,q2;
  input d,clk1,clk2,reset;
  reg q1,q2;

  always @(posedge clk1 or negedge reset)
    if(~reset)q1=1'b0;
  else q1=d;

  always @(posedge clk2)
    q2=clk1&d;//violation location: line no 13.

endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1158: Test Clock Signal "clk1" Used as Data Input.
```

Rule No 1159 Incomplete Sensitivity List

Configurable Parameter

Argument type: none;

Description

The rule checks whether all right-hand-side variables of a combinational always block are included in the sensitivity list.

```
//////////////// Example : Document.v //////////////////////////

module test;
  reg a,b,c;
  always @(a) //warning
    c = a + b;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1159: signal "b" should be included in the sensitivity list.
```

Rule No 1160 Bit Select in Sensitivity List

Configurable Parameter

Argument type: none;

Description

Check to see if there is any bit select of sensitive signal in sensitivity list.

```
/////////////// Example : Document.v ////////////////////////

module test (d, clock, reset, q);
  input d;
  input [1:0] clock, reset;
  output q;
  reg q;
  always @( posedge clock[0] or negedge reset[1] ) //warning here
    if ( reset[1] )
      q <= 1'b0;
    else
      q <= d;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1160: Bit select "reset<1>" in Sensitivity List.
Violated 1160: Bit select "clock<0>" in Sensitivity List.
```

Rule No 1161 Clock Signal Used as Set

Configurable Parameter

Argument type: none;

Description

The rule checks whether there are any clock signals used as set signals.

```
/////////////// Example : Document.v ////////////////////////

module test (q1, q2, clk_1, clk_2, set, d1, d2);
  output q1, q2;
  input clk_1, clk_2, set, d1, d2;
  reg q1, q2;
  wire clk_1, clk_2, set, d1, d2;
  always  @( posedge clk_1 or negedge set )
    if (  ~set )
      q1  <= 1'b1;
    else
      q1  <= d1;
  always  @( posedge clk_2 or posedge clk_1 )
    if (  clk_1 ) //warning
      q2  <= 1'b1;
    else
      q2  <= d2;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1161: Clock Signal "clk_1” Used as Set.
```

Rule No 1162 Non-constant Shift Amount

Configurable Parameter

Argument type: none;

Description

Check to see if there is any non-constant shift amount.

```
/////////////// Example : Document.v ////////////////////////

module test (A, clk, num, b);
  input clk;
  input [7:0] A, num;
  output [7:0] b;
  reg [7:0] b;
  always @(posedge clk)
    b = A << num; //warning on "num", not a static constant
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1162: number of bits to shift ("num<7>") should be a constant.
```

Rule No 1163 Reset Driven by Sequential Logic

Configurable Parameter

Argument type: none;

Description

Check to see if there is any reset signal that is driven by a sequential logic.

```
/////////////// Example : Document.v ////////////////////////

module test (count, c, clk, a, b);
  input clk, a, b;
  output [8:0] count;
  output c;
  reg c;
  reg [8:0] count;
  always @( posedge clk )
    c = a & b; //"c" is output of registe, warning here
  always @(posedge clk)
    begin
    if (c) //"c" is used as reset
      count = 0;
    else
      count = count + 1;
    end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1163: Reset "c" Driven by Sequential Logic.
```

Rule No 1164 Reset Signal Used as Data Input

Configurable Parameter

Argument type: none;

Description

Check to see if there is any reset signal that is also used as a data signal.

```
//////////////// Example : Document.v //////////////////////////

module test (.py(y), .co(count), .cl(clock), .re(reset));
  input clock, reset;
  output [8:0] count;
  reg [8:0] count;
  output y;
  reg y;
  initial
    count <= 0;
  always @(posedge clock)
    begin
    if (reset)//"reset" is reset signal
      count = 0;
    else
      count = count + 1;
    end
  always @(posedge clock)
    y = reset; //warning here, "reset" is used as data input
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1164: Reset Signal "reset" Used as Data Input.

Rule No 1165 Signal Used as Synchronous and

Configurable Parameter

Argument type: none;

Description

Check to see if there is any signal that is used as synchronous and asynchronous set simultaneously.

```
/////////////// Example : Document.v ////////////////////////

module test (y1, y2, data, clock, preset, clear);
  input data, clock, preset, clear;
  output y1, y2;
  reg y1, y2;
  always @(posedge clock or posedge clear or posedge preset)
            //asynchronous set signal "preset"
    begin: forset
    if (clear)
      y1 = 0;
    else
      if (preset)
         y1 = 1;
      else
         y1 = data;
    end
  always @(posedge clock)
    begin
    if (clear)
      y2 = 0;
    else
      if (preset)//synchronous set "preset", mixed
         y2 = 1;
      else
         y2 = data;
    end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1165: Signal "preset" Used as Synchronous and Asynchronous Set.

Rule No 1166 Active Low Signal Name Prefix or Suffix

Configurable Parameter

Argument type: none;

Description

Check to see if the active low signals have recommended prefix or suffix.

```
//////////////// Example : Document.v //////////////////////////

module test (clk,rst,count);
  input clk,rst;
  output [8:0] count;
  reg [8:0] count;
  reg [8:0] nextCount;
  always @(posedge clk or negedge rst)
    if (~rst) //good style for using '_n' for active low reset
      count <= 0;
    else
      count <= nextCount;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1166: active low signal name "rst" does not match to regular expression .*_n.
```

Rule No 1167 Active High Signal Name Prefix or Suffix

Configurable Parameter

Argument type: none;

Description

Check to see if the active high signals have recommended prefix or suffix.

```
/////////////// Example : Document.v ////////////////////////

module test (clk,rst,count);
  input clk,rst;
  output [8:0] count;
  reg [8:0] count;
  reg [8:0] nextCount;
  always @(posedge clk or negedge rst)
    if(rst) //good style for using '_p' for active high reset
      count <= 0;
    else
      count <= nextCount;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1167: active high signal name "rst" does not match to regular expression .*_p.
```

Rule No 1168 Register Output Name Prefix or Suffix

Configurable Parameter

Argument type: none;

Description

Check to see if the register output signals have recommended prefix or suffix.

```
/////////////// Example : Document.v ////////////////////////

module test (clk, datain, dataout);
  input clk, datain;
  output dataout;
  reg dataout;
  always @( posedge clk )
    dataout = datain; //warning on 'dataout', 'dataout_r' recommended
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1168: register ouput name "dataout" does not match to regular expression .*_r.
```

Rule No 1169 Register Input Signal Name Prefix or Suffix

Configurable Parameter

Argument type: none;

Description

Check to see if the register input signals have recommended prefix or suffix.

```
/////////////// Example : Document.v ////////////////////////

module test (clk, datain, dataout);
  input clk, datain;
  output dataout;
  reg dataout;
  always @( posedge clk )
    dataout = datain;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1169: register Input name "datain" does not match to regular expression .*_nxt.
```

Rule No 1170 Instance/Module Name Matches Library Cell Name

Configurable Parameter

Argument type: none;

Description

This rule checks whether a module or an instance name is the same as a library cell name.

```
/////////////// Example : Document.v ////////////////////////

module testini;
   wire a,b,c;
   JANIV JANIV1(a,b,c);
 endmodule
  module JANIV(a,b,c);
   input a,b;
   output c;
   and and1 (c,a,b);
 endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1170: Instance/Module Name "JANIV" uses the same name as an ASIC library cell name.
```

Rule No 1171 Instance Name Length

Configurable Parameter

Argument type: none;

Description

Check to see if the length of the instance name is in the specified range

```
/////////////// Example : Document.v ////////////////////////

module top;
  wire a, b;
  EA u_EA_012345678901234 (a, b); //warning on "u_EA_012345678901234"
endmodule
module EA (a, b);
  input a;
  output b;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1171: The length of Module instance "u_EA_012345678901234" is unconventional and should be in the range from 3 to 16.

Rule No 1172 Identical Module Name and Instance Name

Configurable Parameter

Argument type: none;

Description

Check to see if there is any module instantiation statement in which the instance name and the module name are the same.

```
//////////////// Example : Document.v //////////////////////////

module top (o, a, b);
  input a, b;
  output o;
  test test(o, a, b); //instance name should not be
                    //the same as module name,
                    //"u_test_1" is recommended
endmodule
module test (o, a, b);
  input a, b;
  output o;
  and and1 (o,a,b);
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1172: Instance name "test" should not be the same as the module name.

Rule No 1173 Module instance Name Case

Configurable Parameter

Argument type: none;

Description

Check Module instance Name Case is lower or not.

```
/////////////// Example : Document.v ////////////////////////

module top;
  wire a, b;
  EA U_EA (a, b); //warning on "U_EA"
endmodule
module EA (a, b);
  input a;
  output b;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1173: Module instance "U_EA" should be named in CASE_LOWER case.
```

Rule No 1174 Empty Module

Configurable Parameter

Argument type: none;

Description

The rule checks whether there are any empty modules.

```
/////////////// Example : Document.v ////////////////////////

module test(a);
  input a;
  wire a;
  reg b;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1174: module "test" is empty.
```

Rule No 1175 Gate Name Prefix or Suffix

Configurable Parameter

Argument type: none;

Description

Check to see if the name of gate or primitive instance has a recommended prefix suffix.

```
//////////////// Example : Document.v //////////////////////////

module test (a, b, c);
  input a, b;
  output c;
  and and1(c, a, b); //warning on 'and1'
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1175: gate name "and1<0>" does not match to regular expression gate_.
```

Rule No 1176 Implicit and Confusing Operator Precedence

Configurable Parameter

Argument type: none;

Description

Check to see if there is any suspicious usage of implicit operator precedence.

```
/////////////// Example : Document.v ////////////////////////

module test (f1, f2, f3, a, b, c, d, e, f, g, h, i, j);
  input a, b, c, d, e, f, g, h, i, j;
  output f1, f2, f3;
  reg f1, f2, f3;
  always @(a or b or c or d or e or f or g or h or i or j)
    begin
    if ( a != b & c)
      f1 = a << 2 | b;
    else
      f1 = b << 2 + c;
    f3 = g + h ? i : j;
    end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1176: Implicit and Confusing Operator Precedence on "a".
Violated 1176: Implicit and Confusing Operator Precedence on "b".
Violated 1176: Implicit and Confusing Operator Precedence on "g".
```

Rule No 1177 Reduction Operation on Single-bit Signal

Configurable Parameter

Argument type: none;

Description

Check to see if there is any reduction operation.

```
//////////////// Example : Document.v //////////////////////////

module test (c, a);
  input a;
  output c;
  reg c;
  always @(a)
    c = (&a); //warning on "&a",
              //reduction operation on one bit signal
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1177: reduction operation should not be performed on single-bit signal "a".

Rule No 1178 Gate Instance Found

Configurable Parameter

Argument type: none;

Description

Check to see if there's any instantiated gate.

```
/////////////// Example : Document.v ////////////////////////

module top;
  reg a, b;
  wire c, d, e;
  initial
    begin
    a = 0;
    b = 1;
    #100 $finish;
    end
  always
    #5 a = ~a;
  and U_and(c, a, b); //gate "and" in non-leaf module will cause
                      //systhesizer optimization problem, warning
  test U_test(a, b, d, e);
endmodule
module test (a,b,c,d);
  input a,b;
  output c,d;
  reg c;
  wire d;
  always @(a or b)
    c = ~b;
  and #(3,5) and1(d,a,b);
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1178: Gate Instance "U_and<0>" Found.
Violated 1178: Gate Instance "and1<0>" Found.
```

Rule No 1179 Tri-state Signal Name Prefix or Suffix

Configurable Parameter

Argument type: none;

Description

Check to see if the tri-state output signals have recommended prefix or suffix.

```
//////////////// Example : Document.v //////////////////////////

module smp1(rden, wren, data, tri_out1, cnt1);

  input rden, wren, data;
  output tri_out1;
  output cnt1;

   assign tri_out1 = ( rden ) ? data : 1'bz;
   assign cnt1 = tri_out1 & wren;    //warning here

 endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1179: Tri-state ouput signal name "tri_out1" does not match to regular expression .*_z.
```

Rule No 1180 Multi-bit Expression when One Bit Expression is Expected

Configurable Parameter

Argument type: none;

Description

Check to see if there is any condition expression is wider than single-bit.

```
/////////////// Example : Document.v ////////////////////////

module test (a, c);
  input [1:0] a;
  output c;
  reg c;
  always @(a)
    begin
    if (a) //warning here, one bit expression expected,
           //"a != 2'b00" is desired
      c = 1;
    else
      c = 0;
    end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1180: condition expression on "a" is wider than single-bit.
```

Rule No 1181 Task Not Synthesizable

Configurable Parameter

Argument type: none;

Description

Check to see if there is any task used because it cannot be synthesized.

```
//////////////// Example : Document.v //////////////////////////

module test;
  task Check_0123456789012; //warning on "Check_0123456789012"
  begin
  end
  endtask
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1181: Task "Check_0123456789012" not synthesizable.
```

Rule No 1182 Process Label Prefix or Suffix

Configurable Parameter

Argument type: none;

Description

Check to see if the name of process labels have recommended prefix or suffix.

```
/////////////// Example : Document.v ////////////////////////

module test (clk,rst_p,q);
  inout clk,rst_p;
  output q;
  reg q;
  reg q_nxt;
  always @(posedge clk)
    begin: Dff //a meaningful name is attached on process
      if ( rst_p )
         q = 0;
      else
         q = q_nxt;
    end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1182: process name "Dff" does not match to regular expression *._PROC.
```

Rule No 1183 PROCESS Name Length

Configurable Parameter

Argument type: none;

Description

Check to see if the length of the PROCESS name is in the specified range.

```
//////////////// Example : Document.v /////////////////////////

module test (s, o);
  input s;
  output o;
  reg o;
  always @( s ) begin : P_0123456789012345
                    //warning on "P_0123456789012345"
    o = s;
  end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

Violated 1183: The length of Process "P_0123456789012345" is unconventional and should be in the range from 3 to 16.

Rule No 1184 Process Name Case

Configurable Parameter

Argument type: none;

Description

Check to see if the PROCESS names are all in lower (or upper) case.

```
//////////////// Example : Document.v //////////////////////////

module test (a, inc_dec, sum);
  input a, inc_dec;
  output [7:0] sum;
  reg [7:0] sum;
  always
    begin: COMBINATIONAL_PROC
    if ( inc_dec == 0)
      sum = a ;
    else
      sum = a ;
    end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1184: The Process "COMBINATIONAL_PROC" is unconventional and should be in
CASE_LOWER.
```

Rule No 1185 Unconventional Vector Range Definition

Configurable Parameter

Argument type: none;

Description

The rule checks whether the vector range definition follows the conventional way , such as positive index

```
//////////////// Example : Document.v //////////////////////////

module test(o,a,b,c);
input [8:1]b;
output [9:0]c;
input [0:2] a;
output [3:0] o;
assign o=a;
assign b = c;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1185: descending bit order and zero bound are not used for range declaration of "a".
Violated 1185: descending bit order and zero bound are not used for range declaration of "b".
```

Rule No 1186 Fork Statement Not Synthesizable

Configurable Parameter

Argument type: none;

Description

Check to see if there is any fork statement because it cannot be synthesized.

```
//////////////// Example : Document.v //////////////////////////

module forkjoin1(clk, a, b);
   input clk;
   output a;
   output b;

   reg a, b;

   initial
   begin
      a = 0;
      b = 0;
   end

   always @(posedge clk)
   fork:A
      #2 a = 1;
      #1 b = a;
   join
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1186: fork statement "A" should not be used because it is not synthesizable.
```

Rule No 1187 Wire Not Explicitly Declared

Configurable Parameter

Argument type: none;

Description

Check to see if all wires are declared explicitly.

```
/////////////// Example : Document.v ////////////////////////

module test (a, b, c, d, f);
  input a, b, c, d;
  output f;
  wire f1, f2;
  assign d = f;
  and and1(f1, a, b); //warning on "f1", implicit wire
  or or1(f2, c, d); //warning on "f2", implicit wire
  or or2(f, f1, f2);
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1187: wire "f1" should be explicitly declared.
Violated 1187: wire "f2" should be explicitly declared.
```

Rule No 1188 'tri' Declaration Detected

Configurable Parameter

Argument type: none;

Description

The rule checks whether there are any 'tri' declarations in the design.

```
/////////////// Example : Document.v ////////////////////////

module test(a,b,t);
input a,b;
output t;
tri t;
bufif0 B(t,a,b);
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1188: 'tri' declaration "t" is detected.
```

Rule No 1189 Latch Name Prefix or Suffix

Configurable Parameter

Argument type: none;

Description

Check to see if the latch signal names have recommended prefix or suffix.

```
/////////////// Example : Document.v ////////////////////////

module test (c, a, b);
  input [1:0] a, b;
  output [1:0] c;
  reg [1:0] c;
  always @(a or b)
    if (a)
      c = b; //latch "c" inferred
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1189: latch signal name "c" does not match to regular expression.*_lat.
```

Rule No 1190 Tri-state Buffer Should Not Coexist With Other Logic in a Module

Configurable Parameter

Argument type: none;

Description

This rule checks whether only one tri-state buffer exists in a module at RTL level.

```
/////////////// Example : Document.v ////////////////////////

module smp1(rden, wren, data, tri_out1, cnt1);

  input rden, wren, data;
  output tri_out1;
  output cnt1;

   assign tri_out1 = ( rden ) ? data : 1'bz;
   assign cnt1 = tri_out1 & wren;    //warning here

 endmodul
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1190: tri-state buffer "tri_out1" should not be mixed with other combinational logics in the
module "smp1".
```

Rule No 1191 Inferred Storage Not in Library

Configurable Parameter

Argument type: none;

Description

Check and report all inferred storages not in library.

```
/////////////// Example : Document.v ////////////////////////

module test (clk, d, q);
  input clk, d;
  output q;
  reg q;
  always@(posedge clk)
    q = d; // register inferred not in library
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1191: register inferred on signal "d" not in library.
```

Rule No 1192 Inferred Latch

Configurable Parameter

Argument type: none;

Description

The rule checks whether there are any latches inferred in the design.

```
//////////////// Example : Document.v //////////////////////////

module test (c, a, b);
  input [1:0] a, b;
  output [1:0] c;
  reg [1:0] c;
  always@(a or b)
    if (a)
      c = b; //latch "c" inferred
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1192: latch is inferred on signal "c".

Rule No 1193 Inferred Tri-state

Configurable Parameter

Argument type: none;

Description

The rule checks whether tri-state logic is inferred from the design.

```
/////////////// Example : Document.v ////////////////////////

module test (c, a, b);
  input [1:0] a, b;
  output [1:0] c;
  reg [1:0] c;
  always @(a or b)
    if (a)
      c = b;
    else
      c = 1'bz; //tri-state "c" inferred
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1193: tri-state logic is inferred on signal "c<0>".
Violated 1193: tri-state logic is inferred on signal "c<1>".
```

Rule No 1194 Inferred Mux

Configurable Parameter

Argument type: none;

Description

Check and report all inferred mux logic.

```
/////////////// Example : Document.v ////////////////////////

module test (q, c, a, b);
  input [1:0] a, b, c;
  output [1:0] q;
  reg [1:0] q;
  always@(a or b or c)
    if (a)
      q = b;
    else
      q = c; //a mux "c" inferred
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1194: mux inferred on signal  "q<0>".
Violated 1194: mux inferred on signal  "q<1>".
```

Rule No 1195 Asynchronous Signal Name Prefix or Suffix

Configurable Parameter

Argument type: none;

Description

Check to see if the asynchronous signals have recommended prefix or suffix.

```
//////////////// Example : Document.v //////////////////////////

module test (clk,rst_pa,count);
  input clk,rst_pa;
  output[7:0] count;
  reg[7:0] count;
  reg[7:0] nextCount;
  always @(posedge clk or posedge rst_pa)
    if (rst_pa)
      count <= 0;
    else
      count <= nextCount;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1195: asynchronous signal name "rst_pa" does not match to regular expression .*_a.
```

Rule No 1196 Instance Name Prefix or Suffix

Configurable Parameter

Argument type: none;

Description

Check to see if the name of module instance has a recommended prefix or suffix.

```
//////////////// Example : Document.v //////////////////////////

module top (c,a,b);
  input a, b;
  output c;
  block test_0(c, a, b); //warning on test_0, suggest to use u_test_0
endmodule
module block (c, a, b);
  input a, b;
  output c;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1196: module instance name "test_0" does not match to regular expression U_.*.
```

Rule No 1197 Regular of Active High Signal and Active Low Signal

Configurable Parameter

Argument type: none;

Description

Check to see if there is any active high or low signal name does not follow the regular expression.

```
/////////////// Example : Document.v ////////////////////////

module test;
  wire clk,rst1_n;
  wire rst2, rst_test_n;
  reg [8:0] count;
  always @(posedge clk or negedge rst1_n) begin
    if (~rst1_n)     //warning here
      count = 0;
    else
      count = count + 1;
  end
  always @(posedge clk or negedge rst2) begin
    if (rst2)     //no warning
      count = 0;
    else
      count = count + 1;
  end
  always @(posedge clk or negedge rst_test_n) begin
    if (~rst_test_n) //no warning
      count = 0;
        else
      count = count + 1;
  end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1197: active low reset signal name "rst1_n" does not follow the regular expression rst_.*_n.
```

Rule No 1298 Integer Type Object Detected

Configurable Parameter

Argument type: none;

Description

Check to see if there is any integer type object used.

```
/////////////// Example : Document.v ////////////////////////

module test;
  parameter n=8; //no warning, not an integer
  integer a;      //warning here
  function func1;
  input f1;
  integer f2;     //warning here
    f2=0;
  endfunction
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1198: integer type object "f2" should not be used.
```

Rule No 1199 Next State Name Prefix or Suffix

Configurable Parameter

Argument type: none;

Description

Check to see if the next state register names have recommended prefix or suffix.

```
//////////////// Example : Document.v //////////////////////////

module fsm(out, in, clock, reset);
  output out;
  input in, clock, reset;
  reg out;
  reg [1:0] current, next;
  always @(in or current)
    begin
    out = ~current[1] & current[0];
    next = 0;
    if (current == 0)
      if (in)
         next = 1;
    if (current == 1)
      if (in)
         next = 3;
    if (current == 3)
      if (in)
         next = 3;
      else
         next = 1;
    end
  always @(posedge clock or negedge reset)
    begin
    if (~reset)
      current <= 0;
    else
      current <= next; //warning here on "next"
                 //good style if using "_ns"
                 //as suffix of current state
                 //register "next"
    end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1199: next register name "next<0>" does not match to regular expression .*_ns.
Violated 1199: next register name "next<1>" does not match to regular expression .*_ns.
```

Rule No 1200 Function Returning Real Not Synthesizable

Configurable Parameter

Argument type: none;

Description

Check to see if there is any function returning real type value because it cannot be synthesized.

```
/////////////// Example : Document.v ////////////////////////

module test;
  function real testfun; //warning
    input a, b, cin;
    testfun = a + b + cin;
  endfunction
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1200: function "testfun" returning real type value should not be used because it is not
synthesizable.
```

Rule No 1201 Operation on X/Z

Configurable Parameter

Argument type: none;

Description

Check to see if there is any directly assigned X

```
//////////////// Example : Document.v //////////////////////////

module test (q, clk, reset, d);
  output q;
  input clk, reset, d;
  reg q;
  integer a,b;
  wire clk, reset, d;
  always @(posedge clk)
    q <= d;
  always @(reset)
      assign q = 1'bx;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

Violated 1201: operation on X directly assigned by X should not be used because it will result in simulation mismatch.

Rule No 1202 Operator not Allowed

Configurable Parameter

Argument type: none;

Description

This rule checks whether any operators specified in the argument list are used.

```
/////////////// Example : Document.v ////////////////////////

module test (q, clk, reset, d);
  output q;
  input clk, reset, d;
  reg q;
  integer a,b;
  wire clk, reset, d;
  always @(posedge clk)
    q <= d;
  always @(reset)
      assign q = a * b ;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1202: operator "*" should not be used.
```

Rule No 1203 Assignment is Redundant

Configurable Parameter

Argument type: none;

Description

Check to see if there is any variable assigned in redundant statement.

```
//////////////// Example : Document.v //////////////////////////

module block1( clk, a, b);
input clk, b;
output a;
reg a;
always @(posedge clk) begin
  a <= b;
  a <= ~b;
end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1203: variable "b" assigned here is completely overwrited by following assignments.
```

Rule No 1204 Temporary Variable Used in Nonblocking Assignment

Configurable Parameter

Argument type: none;

Description

Check to see whether temporary variable is used in nonblocking assignment or not.

```
//////////////// Example : Document.v //////////////////////////

module test (clk, y, a, b, c, d);
  output y;
  input a, b, c, d;
  input clk;
  reg y, tmp1, tmp2;
  always @(posedge clk)
  begin
    tmp1 <= a & b;
    tmp2 <= c & d;
    y <= tmp1 | tmp2;
end endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1204: "tmp1"is used like a temporary variable; by non-blocking assignment, it will reduce one
more storage element.
Violated 1204: "tmp2"is used like a temporary variable; by non-blocking assignment, it will reduce one
more storage element.
```

Rule No 1205 Procedural Continuous Assignment Not Synthesizable

Configurable Parameter

Argument type: none;

Description

Check to see if there is any procedural continuous assign statement used because it cannot be synthesized.

```
//////////////// Example : Document.v //////////////////////////

module test (q, clk, reset, d);
  output q;
  input clk, reset, d;
  reg q;
  wire clk, reset, d;
  always @(posedge clk)
    q <= d;
  always @(reset)
    if (~reset)
      assign q = 0; //procedural continuous assignment
                    //non-synthesizable, warning
    else
      deassign q;
endmodule
```

Run Command :

iverilog-0.8 -tlint Document.v

Linting Tool reports:

Violated 1205: procedural continuous assign statement should not be used because it is not synthesizable.

Rule No 1206 Memory is Read and Written at Same Time

Configurable Parameter

Argument type: none;

Description

Check to see if there is a memory being written in one block and, simultaneously, being read in another.

```
/////////////// Example : Document.v ////////////////////////

module test (clock,VMA,R_W,addr,data);
  input clock;
  input VMA;
  input R_W;
  input [1:0] addr;
  inout [1:0] data;
  reg [1:0] macroram [5:0];
  reg [1:0] dataout;
  assign data = R_W ? dataout : 8'hz;
  always @(posedge (clock & VMA))
    begin
      if (R_W == 1)
        dataout=macroram[addr];
      macroram[addr]=data; //"macroram" is read and written at same time
    end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1206: memory written and read at same time.

Rule No 1207 Ignore Don't Care Values in Case Expressions

Configurable Parameter

Argument type: none;

Description

The rule checks whether 'x', 'z' or '?' is used in any case or casex expressions and whether z'or '?' is used in any casez expressions.

```
//////////////// Example : Document.v //////////////////////////

 module top;
        integer ns, ps;
        reg en;

        always
          begin
            casex ({1'b0,1'b?,1'b1 }) //waning here, '?' is used in casex
              3'b?xz: ns = 1;
              3'b00x: ns = 2;
              default: ns = 0;
           endcase
         end

       always
         begin
           casez (en? 1'bz: 1'bz) //warning here, 'z' is used in casez
              1'b?: ps = 1;
              1'b0: ps = 2;
              default: ps = 0;
           endcase
         end
     endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1207: "X/Z/?" is detected in "casex" expression.
Violated 1207: "Z/?" is detected in "casez" expression.
```

Rule No 1208 Reset Signal Feeds into Primary Output

Configurable Parameter

Argument type: none;

Description

Check to see if there is any reset signal that feeds into primary output.

```
//////////////// Example : Document.v /////////////////////////

module test (y, count, clock, reset);
  input clock, reset;
  output [8:0] count;
  reg [8:0] count;
  output y;
  initial
    count <= 0;
  always @(posedge clock or negedge reset)
    begin
    if (reset)
      count = 0;
    else
      count = count + 1;
    end
  assign y = reset;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1208: reset signal should not feed into primary output.

Rule No 1209 Reconverged Clock

Configurable Parameter

Argument type: none;

Description

Check to see if any reconverged clock found.

```
//////////////// Example : Document.v //////////////////////////

module test;
  wire a,b,c,d,d1,f;
  reg y;
  always @(posedge a)
    y=d;
  always @(posedge b)
    y=d;
  always @(posedge c)
    y=d;
  and (b,a,f);
  and (c,a,f);
  and (d1,b,c); //warning here
  always @(posedge d1)
    y=d;
  always @(negedge d1)
    y=d;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1209: reconverged clock "test.d1" found.
```

Rule No 1210 Empty Block

Configurable Parameter

Argument type: none;

Description

The rule checks whether there are blocks without statements.

```
//////////////// Example : Document.v //////////////////////////

module test (clock, q_nxt, q);
  input clock, q_nxt;
  output q;
  reg q;
  always @(posedge clock)
  if(clock)
  begin
  end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1210: empty block detected.
```

Rule No 1211 Separate Clock Generate Circuit in Different Modules

Configurable Parameter

Argument type: none;

Description

Check to see if there is any clock generate circuit mixed with other normal logic.

```
//////////////// Example : Document.v //////////////////////////

module test;
  wire clk, c1, c2;
  reg d, q;
  assign clk = c1 & c2;
  always @(posedge clk)
    q = d;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1211: the clock generate circuit should be separated in different module.
```

Rule No 1212 Flip-Flops with and without Asynchronous Reset/Set Coexist

Configurable Parameter

Argument type: none;

Description

Check to see if any flip-flops with asynchronous reset/set co-exist with flip-flops without asynchronous reset/set in the same always construct

```
//////////////// Example : Document.v /////////////////////////

module smp(RES,CK,FIN1,FIN2,FOUT1,FOUT2);
input      RES,CK,FIN1,FIN2;
output     FOUT1,FOUT2;
reg        FOUT1,FOUT2;
always@(negedge RES or posedge CK)
 if(~RES)
        FOUT1 <= 1'b0;
 else
    begin
        FOUT1 <= FIN1;
        FOUT2 <= FIN2;
    end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

Violated 1212: flip-flop "FOUT1" with asynchronous reset/set and flip-flop "FOUT2" without asynchronous reset/set in the same always construct.

Rule No 1212 Flip-Flops with and without Asynchronous Reset/Set Coexist

Configurable Parameter

Argument type: none;

Description

Check to see if any flip-flops with asynchronous reset/set co-exist with flip-flops without asynchronous reset/set in the same always construct

```
/////////////// Example : Document.v ////////////////////////

module smp(RES,CK,FIN1,FIN2,FOUT1,FOUT2);
input      RES,CK,FIN1,FIN2;
output     FOUT1,FOUT2;
reg        FOUT1,FOUT2;
always@(negedge RES or posedge CK)
 if(~RES)
       FOUT1 <= 1'b0;
 else
    begin
       FOUT1 <= FIN1;
       FOUT2 <= FIN2;
    end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

Violated 1212: flip-flop "FOUT1" with asynchronous reset/set and flip-flop "FOUT2" without asynchronous reset/set in the same always construct.

Rule No 1213 Latch Should Not Coexist with Other Logic in a Module

Configurable Parameter

Argument type: none;

Description

If any latch co-exists with other logic in the same module

```
//////////////// Example : Document.v //////////////////////////

module TriState(y,en,data,Q,G,DATA);
  output y,Q;
  input en,data,G,DATA;
  reg y,Q;
  always @(en or data)
    if(en)
      y = data;
    else
      y = 1'bz;
  always @(G or DATA)
    if(G)
      y <= DATA;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1213: latch "y" should not be mixed with combinational logic.

Rule No 1214 Dangling Else

Configurable Parameter

Argument type: none;

Description

Check to see if there is any "else" part which is associated with the "if" part by virtue of the language's default rule.

```
/////////////// Example : Document.v ////////////////////////

module test (a,b,c);
  input a,b;
  output[2:0] c;
  reg [2:0] c;
  always@(b)
  begin
    if(b)
      if (a)
        c= 1;
      else
       c= 2;
  end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

Violated 1214: the "else" clause should be explicitly associated with the proper "if" clause.

Rule No 1215 Redundant Case label

Configurable Parameter

Argument type: none;

Description

Check to see if there is any Redundant Case label

```
//////////////// Example : Document.v //////////////////////////

module test (y,clock,reset,control);
  input clock,reset,control;
  output [2:0] y;
  reg [2:0] y;
  wire clock,reset,control;
  parameter ST0 = 0,ST1 = 1,ST2 = 2,ST3 = 3;
  reg[1:0] current, next;
  always @(control or current)
    begin
      case (current)
      ST0,ST0: begin y <= 1; next <= ST1; end
        //warning here, redundant case expression
      ST1: begin
              y <= 2;
              if (control) next <= ST2;
              else next = ST3;
           end
      ST2: begin y <= 3; next <= ST3; end
      ST3: begin y <= 4; next <= ST0; end
      default: begin y <= 1; next <= ST0; end
      endcase
  end
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1215: case label "ST0" is redundant.
```

Rule No 1216 Bit Width Mismatch in Comparison of Case Statement

Configurable Parameter

Argument type: none;

Description

Check to see if there is any width mismatch in the comparison of case statement.

```
/////////////// Example : Document.v ////////////////////////

module test;
reg [8:0] out, b;
reg [1:0] a;
reg [1:0] se;
always @( a or b )
  case (a[1:0])
    2'b00, 2'b01:  out = 1;
    2'b10:         out = (b << 2) + 1;
    2'b11:         out = b << 6;
    3'b100:         out = b << 4;
  endcase
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1216: bit width mismatch in the comparison of case statement of "(2)" with "(3)".
```

Rule No 1217 Condition Signal Assigned to 'x' in Default Branch

Configurable Parameter

Argument type: none;

Description

This rule checks whether the signal in a condition expression is assigned to 'x' in the default branch of a case statement.

```
/////////////// Example : Document.v ////////////////////////

module smp(zo,sel);
 input[1:0] sel;
 output[3:0] zo;
 reg [3:0] zo;
 always@( sel )
    case(sel)
     2'b00:     zo = 4'b0001;
     2'b01:     zo = 4'b0010;
     2'b10:     zo = 4'b0100;
     default: zo = 4'bxxxx;
    endcase
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1217: "x" is assigned to default case.
```

Rule No 1218 Non-constant Case Label

Configurable Parameter

Argument type: none;

Description

The rule checks whether there are any non-constant case labels used in case statements.

```
//////////////// Example : Document.v //////////////////////////

module test (out, sel, tmp, data);
  output out;
  input [2:0] sel, data;
  input [1:0] tmp;
  reg out;
  always @(sel or data or tmp)
    casex (sel) // Synopsys full_case
      3'bx0: out = data[0];
      3'b01: out = data[1];
      tmp:    out = data[2]; //warning
      default: out = 0;
    endcase
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1218: case label "tmp" is not a constant.
```

Rule No 1219 Unreachable Default Branch of Case Statement

Configurable Parameter

Argument type: none;

Description

The rule checks if all the possibilities of the case expression are covered by case alternatives.

```
//////////////// Example : Document.v //////////////////////////

module test(foo, bar, bar_0, bar_1, bar_2, bar_3);
input [5:0] foo;
input [35:0] bar_0, bar_1, bar_2, bar_3;
output [35:0] bar;
reg [35:0] bar;
always @( foo or bar_0 or bar_1 or bar_2 or bar_3 ) begin
  case (foo[1:0])
    2'd0 : bar = bar_0;
    2'd1 : bar = bar_1;
    2'd2 : bar = bar_2;
    2'd3 : bar = bar_3;
    default : bar = 36'd0; //warning here
  endcase
end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

Violated 1219: default branch is unreachable because case alternatives have covered all the possibilities of case expression.

Rule No 1220 Incomplete Case Expression with Default Clause and Synopsys Full_case Directive

Configurable Parameter

Argument type: none;

Description

The rule checks whether all possible values of a case expression are covered in a case where a default clause and "synopsys full_case" directive exists.

```
//////////////// Example : Document.v //////////////////////////

module test;
  wire [1:0] a;
  reg b;
  always @(a)
    casez (a[1:0]) // synopsys full_case
      2'b00: b = 1'b1;
      2'b01: b = 1'b0;
      2'b10: b = 1'bx;
      default: b = 1'bz;  //2'b11 is not covered but default exists.
  endcase
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

Violated 1220: not all possible values of the case expression are covered in a full case but a default clause exists

Rule No 1221 Case Statement Not Fully Specified

Configurable Parameter

Argument type: none;

Description

The rule checks whether all possible branches of the case expression are specified.

```
/////////////// Example : Document.v ////////////////////////

module test (out0, in1, in2, in3, sel);
  input [1:0] in1, in2, in3, sel;
  output [1:0] out0;
  reg [1:0] out0;
  always @(in1 or in2 or sel)
    case (sel) //warning
      2'b00: out0 = in1;
      2'b01: out0 = in2;
      2'b10: out0 = in3;
    endcase
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1221: not all possible case branches are specified.
```

Rule No 1222 Incomplete Case Expression with Default Clause

Configurable Parameter

Argument type: none;

Description

When case labels do not cover all possible values of case expression, default clause could be exercised.

```
/////////////// Example : Document.v ////////////////////////

module test;
        wire [1:0] a;
        reg b;

        always @(a)
          casez (a[1:0])   //warning here, 2'b11 is not covered but default
            2'b00: b = 1'b1;
            2'b01: b = 1'b0;
            2'b10: b = 1'bx;
           default: b = 1'bz;
       endcase
     endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

Violated 1222: not all possible values of the case expression are covered but a default clause exists

Rule No 1223 Latch to Latch Connected should be Enabled in Different Phase

Configurable Parameter

Argument type: none;

Description

Check to see that connection between latches should be enabled in different phase.

```
/////////////// Example : Document.v ////////////////////////
 
module top (a3);
  output a3;
  wire [1:0] tri_,latch;
  wire c;
  wire a1;
  sub s1(latch[0], c, a2);
  assign a1=a2;
  sub s2(latch[0], a1, a3);
endmodule
module sub (latch , c, q);
  input latch, c;
  output q;
  wire latch, c;
  reg q;
  always @(latch or c) begin
    if(latch)
    begin
      q=c;
    end
  end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1223: latch to latch should be enabled in different phase.
```

Rule No 1224 Bit Width Mismatch in Bitwise Operation

Configurable Parameter

Argument type: none;

Description

The rule checks whether any bit widths mismatch between operands of a bitwise operation.

```
//////////////// Example : Document.v //////////////////////////

module test (result, a, b);
  input [2:0] a;
  input [3:0] b;
  output [3:0] result;
  reg [3:0] result;
  always @(a or b)
    result = a & b; //warning on "a" and "b"
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1224: bit width of operand "a(3)" does not match that of operand "b(4)" in bitwise operation.
```

Rule No 1225 Possible Loss Value in Multiplication

Configurable Parameter

Argument type: none;

Description

Check to see if there is any possible value loss in multiplication.

```
//////////////// Example : Document.v //////////////////////////

module test;
  wire [7:0] a;
  wire [7:0] b;
  wire [7:0] c;
  assign a = b * c; //warning on "b" and "c"
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1225: possible loss value between left operand "b" and right operand "c" in Multiplication.

Rule No 1226 Possible Loss of Carry or Borrow in Addition

Configurable Parameter

Argument type: none;

Description

Check to see if there is any possible loss of carry or borrow in addition or subtraction.

```
/////////////// Example : Document.v ////////////////////////

module test;
  wire [7:0] a;
  wire [7:0] b;
  wire [1:0] c;
  assign a = b + c;  //warning on "b" and "c"
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1226: possible loss of carry in addition left operand and right operand.

Rule No 1227 Operand Bit Size Mismatch in Addition or Subtraction

Configurable Parameter

Argument type: none;

Description

The rule checks whether bit width mismatches occur between operands of addition or subtraction.

```
/////////////// Example : Document.v ////////////////////////

module test;
  wire [7:0] a;
  wire [7:0] b;
  wire [1:0] c;
  assign a = b + c;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1227: bit width of left-hand-side operand does not match the right-hand-side operand in
"Addition".
```

```
//////////////// Example : Document.v /////////////////////////

module test;
  wire [7:0] a;
  wire [7:0] b;
  wire [1:0] c;
  assign a = b - c;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1227: bit width of left-hand-side operand does not match the right-hand-side operand in
"Subtraction".
```

Rule No 1228 Variables with Different Bit Widths Used in Conditional Assignment Branches

Configurable Parameter

Argument type: none;

Description

The rule checks whether operands in conditional assignment are of the same size.

```
/////////////// Example : Document.v ////////////////////////

module test (a, b, c, d);
  input b;
  input [2:0] c;
  input [3:0] d;
  output [3:0] a;
  assign a = b ? c[2:1] : d[3:0];
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

Violated 1228: the bit widths of the conditional assignment operands are different.

Rule No 1229 All Bits Shifted Out

Configurable Parameter

Argument type: none;

Description

Check to see if there is any shift operation that has all the bits in the signal shifted out.

```
//////////////// Example : Document.v ///////////////////////////

module test (A, clk, num, b);
  input clk;
  input [7:0] A;
  input [7:0] num;
  output [7:0] b;
  reg [7:0] b;

  always @(posedge clk)
    b = A << num;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint -S Document.v
```

Linting Tool reports:

```
Violated 1229: All Bits of signal "A" shifted Out.
```

Rule No 1230 Function Called in an Always Block

Configurable Parameter

Argument type: none;

Description

This rule checks whether there are any functions called in an always block

```
/////////////// Example : Document.v ////////////////////////

module smp(CK,A,B,C,D,G,H,ZF);
input   CK;
input   A,B,C,D,G,H;
output  ZF;
reg     ZF;
function ZFC;
input A,B,C,D,G,H;
 begin
    case ({G,H})
     2'b00 :   ZFC = A;
     2'b01 :   ZFC = B;
     2'b10 :   ZFC = C;
     2'b11 :   ZFC = D;
     default : ZFC = 1'bx;
    endcase
 end
endfunction
always@( posedge CK )
    ZF <= ZFC ( A,B,C,D,G,H );
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1230: function "ZFC" is called in an always block.
```

Rule No 1270 'iff' Construct Not Synthesizable

Configurable Parameter

Argument type: none;

Description

The rule checks whether there are any 'iff' constructs, which are not synthesizable, used in the design.

Example

```
///////////////example : document.v////////////

module test;
  reg rst, d;
  reg q;
  always @ (d iff rst == 0) // Worning Here
  begin
    if ( ~rst )
      q = 1'b0;
    else
      q = d;
  end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

Violated 1270: 'iff' construct is not synthesizable.

Rule No 1271 Keyword 'changed' Not Synthesizable

Configurable Parameter

Argument type: none;

Description

The rule checks whether there is keyword 'changed' used. The construct is not synthesizable.

Example

```
/////////////example : document.v////////////

module test;
  reg clk, rst, d;
  reg q;
  always @ (changed rst, posedge clk or d)  // Worning Here
  begin
    if ( ~rst )
      q = 1'b0;
    else
      q = d;
  end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1271: Keyword 'changed' is not synthesizable.
```

Rule No 1272 Arithmetic/Relational Operations Sharing with Large Operand Not Allowed

Configurable Parameter

Argument type: integer;
Argument description: specify the maximum bit width number;
Default value: "8" for Verilog;
Argument Value: "16";

Description

This rules check whether either the arithmetic operator (+, -, *) or relational operator (<, >, =) are shared with a large bit-width operand. A violation will be reported if the operator is shared and if the bit-width of the operand exceeds the specified argument.

Example

```
/////////////example : document.v///////////

module smp(IN1, IN2, OUT1, OUT2);
  input [16:0] IN1;
  input [16:0] IN2;
  output [16:0] OUT1;
  output [16:0] OUT2;
  reg [16:0] OUT1;
  reg [16:0] OUT2;
  always @(IN1 or IN2)
    if(IN1 < IN2)
      OUT1 = IN1 + 2; //warning here if width argument is set to 16
    else
      OUT1 = IN1 - 2; //warning here if width argument is set to 16
   always @(IN1 or IN2)
    if(IN2 == IN1)
      OUT2 = IN2 * 4; //warning here if width argument is set to 16
    else
      OUT2 = IN2 / 4; //warning here if width argument is set to 16
endmodule
```

Run Command :

```
iverilog-0.8 -tlint –S Document.v
```

Linting Tool reports:

Violated 1272: Arithmetic/Relational Operations Sharing with Large Operand Not Allowed. It Exceed imit 16 Bit.

Rule No 1273 Large Multiplier Inferred

Configurable Parameter

Argument type: integer;
Argument description: specify the maximum number of bit-width for inferred multipliers;
Default value: "16" for Verilog;
Argument Value: "16";

Description

This rule checks whether the bit width of inferred multipliers at RTL level is within the specified limit. A violation will be reported when the bit width exceeds the specified argument value.

Example

```
/////////////example : document.v///////////

module smp(IN1, IN2, OUT1, OUT2);
  input [16:0] IN1;
  input [15:0] IN2;
  output [16:0] OUT1;
  output [15:0] OUT2;
  reg [16:0] OUT1;
  reg [15:0] OUT2;

always @(IN1 or IN2)
    if(IN1 < IN2)
      OUT1 = IN1 * 2; //warning here if width argument is set to 16
    else
      OUT1 = IN1 / 2;

always @(IN1 or IN2)
    if(IN2 > IN1)
      OUT2 = IN2 * 4; //no warning here
    else
      OUT2 = IN2 / 4;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint –S Document.v
```

Linting Tool reports:

```
Violated 1273: Do not infer large multiplier by the operand "smp.IN2" whose bit-width is larger than 16.
```

Rule No 1274 Use always_comb to Model Combinational Behavior

Configurable Parameter

Argument type: none;

Description

This rule checks whether always @* is used to model combinational logic; always_comb is the recommended alternative.

Example

```
//////////////example : document.v////////////

module test;
  reg a1, a2, a, b, c;
  reg clk, sel, s;
  reg q, q1, q2, d, d1;
  always @* begin //always_comb preferred;
    a = b - c;
    a = b * c;
    a = b / c;
    a = b % c;
  end
  always_ff @( posedge clk ) begin //no warning, since it is inferred as a
    q <= d;
  end
  always_latch @* //no warning, since it is inferred as a latch;
    if ( sel )
      q1 <= d1;
  always_ff @* //always_comb preferred;
    case (s)
      1: q2 = a1;
      0: q2 = a2;
    endcase
endmodule
```

Run Command :

```
iverilog-0.8 -tlint –S Document.v
```

Linting Tool reports:

Violated 1274: Use "always_comb" to model combinational behavior.

Rule No 1275 'always_ff' Not Used for Sequential Blocks

Configurable Parameter

Argument type: none;

Description

The rule checks whether there are any sequential blocks modeled by an always block. The recommended construct is 'always_ff'.

Example

```
//////////////example : document.v////////////

module test;
  reg clk, sel;
  reg a,b,c,d,d1;
  reg q, q1;

  always @* begin //no violation because it is inferred as combination logic
    a = b & c;
  end

  always @( posedge clk ) begin //report warning
    q1 <= d1;
  end

  always @* begin //no warning, since it is inferred as a latch;
    if ( sel )
      q <= d;
  end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint –S Document.v
```

Linting Tool reports:

```
Violated 1275: "always_ff" is not used to model sequential blocks.
```

Rule No 1276 'always_latch' Not Used for Sequential Blocks

Configurable Parameter

Argument type: none;

Description

The rule checks whether there are any sequential blocks modelled by an always block. The recommended construct is 'always_latch'.

Example

```
//////////////example : document.v////////////

module test;
  reg clk, sel;
  reg a,b,c,d,d1;
  reg q, q1;

  always @* begin //no violation because it is inferred as combination logic
    a = b & c;
  end

  always @* begin //report warning
    if ( clk )
    q1 <= d1;
  end

  always_latch @* begin //no warning, since it is inferred as a latch;
    if ( sel )
      q <= d;
  end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint –S Document.v
```

Linting Tool reports:

Violated 1276: "always_latch" is not used to model sequential blocks.

Rule No 1277 If Statement Describing Asynchronous Set/Reset should be the Most Beginning Statement in a Always Block

Configurable Parameter

Argument type: none;

Description

Report warning if a if-statement describing asynchronous set or reset is not in the beginning of always block or report warning if any other logic statement in always block both before and after asynchronous set/reset.

Example

```
/////////////example : document.v///////////

module test(d1, clk, R, d, q, d2);
  input d1, clk, R, d;
  output q, d2;
  reg q, d2;

  always @( posedge clk or posedge R) begin
    d2 = d1;
    if (R) // not the first statement
      q = 0;
    else
      q = d;
  end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint –S Document.v
```

Linting Tool reports:

Violated 1277: If statement describing asynchronous set/reset should be put before any other logic statement in always block.

Rule No 1278 Use Simple Expression for Asynchronous Set or Reset

Configurable Parameter

Argument type: none;

Description

The rule checks whether there are any prohibited operations used for asynchronous set or reset expressions in if-statements.

Example

```
//////////////example : document.v////////////

module test2(d1, clk, R, d, q, d2);
  input d1, d2, clk, R, d;
  output q;
  reg q;
  always @( posedge clk or posedge R) begin
    if ( | R ) //not a simple condition
      q = 0;
    else
      q = d;
  end
  always @( posedge clk or posedge R) begin
    if ( & R ) //not a simple condition
      q = 0;
    else
      q = d;
  end
  always @( posedge clk or posedge R) begin
    if (R == 0) //not a simple condition
      q = 0;
    else
      q = d;
  end
  always @ (posedge clk or negedge R)
    begin
      if (R) // No Warning
         q = 0;
      else
         q = d;
    end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint –S Document.v
```

Linting Tool reports:

```
Violated 1278: Don't use complex expression in the condition expression for asynchronous set or reset.
```

Rule No 1279 Inputs of a Tri-state Bus Not Generated from a Single Clock Source

Configurable Parameter

Argument type: none;

Description

The rule checks whether all the inputs of a tri-state bus are generated from a single clock source.

Example

```
//////////////example : document.v////////////

module test (clk1, clk2, en1, en2, d1, d2, q, qq);
  input clk1, clk2, en1, en2, d1, d2;
  output q, qq;
  wire clk1, clk2, d1, d2, q, en1, en2;
  reg q1, q2;

  always @(posedge clk1)
    q1 = d1;

  always @(posedge clk2)
    q2 = d2;

  assign q = q1 ? en1 : 'bZ;
  assign qq= q2 ? en2 : 'bZ;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint –S Document.v
```

Linting Tool reports:

```
Violated 1279: The inputs of the tri-state bus "qq" are not generated from any clock source.
Violated 1279: The inputs of the tri-state bus "q" are not generated from any clock source.
```

Rule No 1280 Tri-state Buffer in a Clock Path

Configurable Parameter

Argument type: none;

Description

This rule checks whether there are tri-state buffers in a clock path.

Example

```
//////////////example : document.v////////////

module test (clk, en, d, q);
  input clk, en, d;
  output q;
  wire clk1, clk, d, en, clk2;
  reg q;

  always @(posedge clk)
    q = d;

 always @(posedge clk1)
    q = d;

  assign clk1 = en ? clk : 'bZ;

  always @(posedge clk2)
    q = d;

  assign clk2 = en ? clk : 'bZ;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint –S Document.v
```

Linting Tool reports:

```
Violated 1280: TriState exists in a clock path from "clk" to "clk2".
Violated 1280: TriState exists in a clock path from "clk" to "clk1".
```

Rule No 1281 Both Set and Reset Found for a Flip-flop

Configurable Parameter

Argument type: none;

Description

Check to see if a flip-flop/latch has both set and reset

Example

```
//////////////example : document.v////////////

module test(clk,set,reset,set1,reset1,enable,D,Q1,Q2);
  input clk,set,reset,set1,reset1, enable;
  input D;
  output Q1,Q2;
  reg Q1,Q2;

always @(posedge clk or negedge reset or posedge set)
  begin
    if (set)
      Q1 <=1'b1;
    else if (!reset) //violation here
      Q1 <= 1'b0;
    else
      Q1 <= D;
  end

always @(posedge clk or negedge reset1 or posedge set1)
  begin
    if (set1)
      Q2 <= 1'b1;
    else if(!reset1)
      Q2 <= 1'b0; //violation here
    else
      Q2 <= D;
  end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint –S Document.v
```

Linting Tool reports:

```
Violated 1281: Set (set1) and reset (reset1) are specified for the same flip-flop/latch.
Violated 1281: Set (set) and reset (reset) are specified for the same flip-flop/latch.
```

Rule No 1282 Always Block Contains Multiple Resets

Configurable Parameter

Argument type: none;

Description

This rule checks whether there is only one reset in an always block. A violation is reported if the number of resets exceeds one in an always block.

Example

```
//////////////example : document.v////////////

module smp(RES1,RES2,CK,FIN,FOUT);
  input     RES1,RES2,CK,FIN;
  output    FOUT;
  reg       FOUT;
  parameter D1 = 1;

  always@(negedge RES1 or negedge RES2 or posedge CK)
    if(RES1 || !RES2)
      FOUT <= #D1 1'b0;
    else
      FOUT <= #D1 FIN;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint –S Document.v
```

Linting Tool reports:

```
Violated 1282: Multiple asynchronous resets are specified in the same always construct.
```

Rule No 1283 Dual Set or Reset Detected

Configurable Parameter

Argument type: none;

Description

Check to see if there is any dual set or reset in edge-sensitive logic.A violation is reported if the number of resets exceeds one in edge-sensitive logic.

Example

```
//////////////example : document.v////////////

module test (clock, ctrl1, ctrl2, q, q1, d);
  input clock, ctrl1, ctrl2;
  input d;
  output q, q1;
  reg q, q1;

  always @( posedge clock or negedge ctrl1 or negedge ctrl2 )
    if ( !ctrl1 )
      q <= #1 1'b0; //ctrl1 is treated as reset
    else if ( !ctrl2 )
      q <= #1 1'b0; //ctrl2 is treated as reset, warning here
    else
      q <= d;
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1283: Signal "q" has already been reset.
```

Rule No 1284 End Point Not Generated from a Single Clock Source

Configurable Parameter

Argument type: none;

Description

This rule checks whether there are any end points generated from more than one clock source tree.

Example

```
//////////////example : document.v////////////

module test (clk1, clk2, clk, d1, d2, q, qq, qqq);
  input clk1, clk2, clk, d1, d2;
  output q, qq, qqq;
  wire temp, temp1, temp2;
  reg  q1, q2, q, qq, qqq;

  always @(posedge clk1)
    q1 = d1;

  always @(posedge clk2)
    q2 = d2;

  assign temp2 = q1 & q2;
  assign temp1 = temp2;
  assign temp = temp1;

/////  FOR LPM  /////
  always @(negedge clk)begin
    qq = temp;  // "temp" comes from more than one clock source.
    q = q1 & q2;  // out put of "q1" & "q2" comes from more than one clock source.
  end

/////  FOR Signal  /////
  always @(q1 or q2)begin
      qqq = temp;  // "temp" comes from more than one clock source.
  end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint –S Document.v
```

Linting Tool reports:

```
Violated 1284: The input of register "q" comes from more than one clock source.
Violated 1284: The input of register "qq" comes from more than one clock source.
Violated 1284: The input of register "qqq" comes from more than one clock source.
```

Rule No 1285 A Signal is Connected to Both Input and Output Ports of an Instance

Configurable Parameter

Argument type: none;

Description

For a module instance, compare each output signal in its high-connection expression with each input signal. If there is a signal connected to both input and output port of the instance, report warning

Example

```
//////////////example : document.v////////////

module test( m, n, o );
  input m, n;
  output o;
  assign o = m & n;
endmodule

module test1( x, y, z );
  input x, y;
  output z;
  assign z = x | y;
endmodule

module test2( a, b, c );
  input a, b;
  output c;
  test u1( c, b, c ); //here signal connected to output "c" goes into
                      //input "a" of test1 again.
  //or something like following
  test1 u2( a&c, b, c);
endmodule
```

Run Command :

```
iverilog-0.8 -tlint –S Document.v
```

Linting Tool reports:

```
Violated 1285: Signal "c" should not connect both input and output port of the instance.
Violated 1285: Signal "o" should not connect both input and output port of the instance.
Violated 1285: Signal "z" should not connect both input and output port of the instance.
```

Rule No 1286 Bus Direction Consist on Port Binding

Configurable Parameter

Argument type: none;

Description

Bus direction should keep consistancy between port instance and port on module binding.

Example

```
//////////////example : document.v////////////

module test(in1,a);
  input [3:6] in1;
  input a;
endmodule

module top ();
  wire [6:4] B;
  test i_test(B, B[4]);
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1286: Bus direction of "top.B" is not consistent with "top.i_test.in1".
Violated 1286: Bus direction of "top.i_test.in1" is not consistent with "top.B".
```

```
//////////////example : document.v////////////

module m1(a,b,c);
  input [0:7] a;
  input [7:0] b;
  output [7:0] c;
endmodule
module m2(.a({a0,a1}),b,c);
  input [0:3] a0;
  input [3:0] a1;
  input [7:0] b;
  output [7:0] c;
endmodule
//module instantiation:
module top();
  wire [7:0] A;
  wire [7:0] B;
  wire [3:0] C1;
  wire [0:3] C2;
  wire [0:1] C3;
  wire [1:0] C4;
  m1 u1_m1(.a(A), //inconsistant,report warning on A vs a
        //since it is the case "normal portinst" + "normal port"
  .b(B),
  .c({C1,C2})); //inconsistant ,report warning on C2 vs c
              //it is "simple concat portinst" + "normal port"
  m1 u2_m1(.a(A),//inconsistant,report warning on A vs a
  .b(B),
  .c({C1,{C3,C4}}));//inconsistant ,report warning on C3 vs c
                  //it is "complex concat portinst"+"normal port"
  m2 u_m2(.a(A),//inconsistant,report warning on A vs a0
              //it is "normal portinst"+"concat port"
  .b(B),
  .c({C1,C2}));//inconsistant ,report warning on C2 vs c
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1286: Bus direction of "top.A" is not consistent with "top.u2_m1.a".
Violated 1286: Bus direction of "top.A" is not consistent with "top.u1_m1.a".
Violated 1286: Bus direction of "top.A" is not consistent with "top.u_m2.a0".
Violated 1286: Bus direction of "top.C2" is not consistent with "top.u_m2.c".
Violated 1286: Bus direction of "top.C2" is not consistent with "top.u1_m1.c".
Violated 1286: Bus direction of "top.C3" is not consistent with "top.u2_m1.c".
Violated 1286: Bus direction of "top.u1_m1.a" is not consistent with "top.u_m2.a1".
Violated 1286: Bus direction of "top.u1_m1.a" is not consistent with "top.A".
Violated 1286: Bus direction of "top.u1_m1.c" is not consistent with "top.C2".
Violated 1286: Bus direction of "top.u2_m1.a" is not consistent with "top.u_m2.a1".
Violated 1286: Bus direction of "top.u2_m1.a" is not consistent with "top.A".
Violated 1286: Bus direction of "top.u2_m1.c" is not consistent with "top.C3".
Violated 1286: Bus direction of "top.u_m2.a0" is not consistent with "top.A".
Violated 1286: Bus direction of "top.u_m2.a1" is not consistent with "top.u2_m1.a".
Violated 1286: Bus direction of "top.u_m2.a1" is not consistent with "top.u1_m1.a".
Violated 1286: Bus direction of "top.u_m2.c" is not consistent with "top.C2".
```

```
/////////////example : document.v///////////

module test(.a({in1,in2}));
  input [0:3] in1;
  input [3:0] in2;
endmodule
module top ();
  wire [0:3] B;
  wire [3:0] C;
  test i_test({C,B});//will not check the portinst,
                 //since both port and portinst are concated,
                 //though C[3:0]<==>in1[0:3], B[0:3]<==>in2[3:0]
                 //which break the the rule.
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1286: Bus direction of "top.B" is not consistent with "top.i_test.in2".
Violated 1286: Bus direction of "top.C" is not consistent with "top.i_test.in1".
Violated 1286: Bus direction of "top.i_test.in1" is not consistent with "top.C".
Violated 1286: Bus direction of "top.i_test.in2" is not consistent with "top.B".
```

```
//////////////example : document.v////////////


module m1(a,b,c);
  input [0:7] a;
  input [7:0] b;
  output [7:0] c;
endmodule

module m2(a,b,c);
  input a;
  input [7:0] b;
  output [5:5]c;
endmodule

//module instantiation:
module top();
  wire [7:0] A;
  wire [7:0] B;
  wire [3:0] C1;
  wire [0:3] C2;

  m2 u_m2(.a(A),//not report on a single bit port.
  .b(B),
  .c({C1,C2}));//not report on a single bit port.

  m1 u3_m1(.a(A&B), //not report when introduce a rtl inst. (A&B).
  .b(B),
  .c(C1)); //not report when introduce a rtl inst (C2).
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

Rule No 1287 Bit Width Mismatch in Logical Operation

Configurable Parameter

Argument type: none;

Description

This rule checks whether the signal bit width for logical operations are equivalent. A violation will be reported if the bit-width mismatches on the left/right side of the logical operator.

Example

```
//////////////example : document.v////////////

module smp(A,B,X,Y,C,Z);
  input X;
  input [1:0]Y;
  input [1:0] A;
  input [2:0] B;
  output C,Z;
  assign C = ( A || B ) ? 1'b1 : 1'b0; //warning on (A && B) here
  assign Z = ( X && Y ) ? 1'b1 : 1'b0; //warning on (X && Y) here
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1287: Bit Width "A" and "B" Mismatch in Logical Operation.
Violated 1287: Bit Width "X" and "Y" Mismatch in Logical Operation.
```

Rule No 1288 Large Bit Width of Reduction Operand

Configurable Parameter

Argument type: integer;
Argument description: specify the maximum bit width for the reduction operand;
Default value: "32" for Verilog;
Argument Value: "32";

Description

This rule checks whether the bit width of reduction operands is within the specified limit. A violation will be reported when the bit-width exceeds the specified argument value.

Example

```
/////////////example : document.v////////////

module test;
  reg [39:0] a,d;
  wire b,e;
  wire [31:0] f;
  assign b = &a;  //  Warning on a;
  assign e = |f;  //  No Warning
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports following if the argument value is ("32"):

```
Violated 1288: The bit width of reduction operand "a" is larger than 32.
```

Rule No 1289 Signal Assigned to 'x'

Configurable Parameter

Argument type: none;

Description

This rule checks whether the signal is assigned to 'x'. A violation will be reported if a signal assigned to ‘x'.

Example

```
//////////////example : document.v////////////

module smp(zo,fout,sel,res,ck,fin);
  input[1:0] sel;
  input        res, ck, fin;
  output[3:0] zo;
  output       fout;
  reg [3:0] zo;
  reg fout;
  parameter D1=1;
  always@( sel )
    case(sel)
      2'b00:     zo = 4'b0001;
      2'b01:     zo = 4'b0010;
      2'b10:     zo = 4'b0100;
      default: zo = 4'bxx00;    //  Warning Here
    endcase
  always@( negedge res or posedge ck )begin
    if( ~res )
      fout <= #D1 1'b0;
    else if(zo==4'b0100)
      fout <= #D1 1'bz;     //  Warning Here
    else
      fout <= #D1 fin;
  end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint –S Document.v
```

Linting Tool reports:

```
Violated 1289: Signal smp._s30 Assigned to 'x'.
Violated 1289: Signal smp._s31 Assigned to 'x'.
Violated 1289: Signal smp._s39 Assigned to 'x'.
Violated 1289: Signal smp._s40 Assigned to 'x'.
```

Rule No 1290 User Defined Primitive should be Named in Lower Case

Configurable Parameter

Argument type: none;

Description

Check to see if user defined primitive all named and veriable are in lower case.

Example

```
//////////////example : document.v////////////

module test(cLK,carryout,carryin, ain, BIN);
  output carryout,cLK;
  input carryin, ain, BIN;
  CarrY u1( carryout, carryin, ain, BIN );
  not a (cLK,carryout);
endmodule

primitive CarrY(carryout,carryin,ain,bin); //Warning here
  output carryout;
  input carryin, ain,bin;
  table
// carryIn aIn bIn carryOut
      0     0   0 :  0;
      0     0   1 :  0;
      0     1   0 :  0;
      0     1   1 :  1;
      1     0   0 :  0;
      1     0   1 :  1;
      1     1   0 :  1;
      1     1   1 :  1;
  endtable
endprimitive
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1290: UDP Signal Name "BIN" should be named in lower case.
Violated 1290: UDP Name "CarrY" should be named in lower case.
```

Rule No 1292 Instance Name Related to Module Name

Configurable Parameter

Argument type: none;

Description

Instance names must be related to the module name and indexed by an integer if multiple instantiated.

Example

```
/////////////example : document.v///////////

module adderx(ss);
  input [1:0] ss;
  parameter check = 1;
endmodule

module adderx1(ss);
  input [1:0] ss;
  parameter check = 1;
endmodule

module test_adder;
  wire [1:0] in1;

  adderx adderx_1(in1);         // OK
  adderx adderx_2(in1);         // OK
  adderx test_1_adderx(in1);      // OK
  adderx testadder(in1);       // Waring,instname does not relate to module name.
                                 // Warning,instname does not have an index.
  adderx adderx_last(in1);      // Warning,instname does not have an index.

  adderx T111111_eSt_tEsT_11_(in1);  // Waring,instname does not relate to module name.
  adderx adder_last_1(in1);      // Waring,instname does not relate to module name.
  adderx a_0001A_1 (in1);        // Waring,instname does not relate to module name.

  adderx1 adderx1a_1(in1);        // OK
  adderx1 adDabcerxd_0(in1);       // Waring,instname does not relate to module name.
  adderx1 AdDabcerxd2(in1);        // Waring,instname does not relate to module name.
                              // Warning,instname does not have an index.
  adderx1 addabcerxe1(in1);       // Waring,instname does not relate to module name.
                              // Warning,instname does not have an index.
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1292: Instance name "testadder" must be related to module Name.
Violated 1292: Instance name "testadder" indexed by an integer followed by a '_'.
Violated 1292: Instance name "adderx_last" indexed by an integer followed by a '_'.
Violated 1292: Instance name "T111111_eSt_tEsT_11_" must be related to module Name.
Violated 1292: Instance name "adder_last_1" must be related to module Name.
Violated 1292: Instance name "a_0001A_1" must be related to module Name.
Violated 1292: Instance name "adDabcerxd_0" must be related to module Name.
Violated 1292: Instance name "AdDabcerxd2" must be related to module Name.
Violated 1292: Instance name "AdDabcerxd2" indexed by an integer followed by a '_'.
Violated 1292: Instance name "addabcerxe1" must be related to module Name.
Violated 1292: Instance name "addabcerxe1" indexed by an integer followed by a '_'.
```

Rule No 1293 Module Name Follow Predefined Pattern

Configurable Parameter

Argument type: string;
Argument description: specify the pattern for checking;
Default value: ".*GTECH.*" for Verilog;
Argument Value: ".*GTECH.*";

Description

Report warning if a module (not top) is named in a predefined pattern,reporting location is at the instance of the module.

Example

```
//////////////example : document.v////////////

module mGTECH1oo1(i, o);
  input i;
  output o;
endmodule

module mGTECH1(i, o);
  input i;
  output o;
  mGTECH1oo1 inst_2 (i, o); //warning with the pattern .*GTECH.*
endmodule

module GTECHtest; // No warning Here as Top Module
  mGTECH1 inst_1 (i, o); //warning with the pattern .*GTECH.*
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1293: Module "mGTECH1" (of instance "inst_1") is named in the predefined pattern".*GTECH*.
Violated 1293: Module "mGTECH1oo1" (of instance "inst_2") is named in the predefined pattern
".*GTECH*.
```

Rule No 1294 Task in 'always_comb' Block

Configurable Parameter

Argument type: none;

Description

The rule check whether tasks are used inside always_comb blocks. A violation will be reported if 'task' is used in 'always_comb' block

Example

```
//////////////example : document.v////////////

module comb1(input a, b ,c, output reg [1:0] y);
  task ort1;
    input a;
    y[0] = a | b | c;
  endtask

  always_comb ort1(a); //warning here
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1294: Task is used in "always_comb" block.
```

Rule No 1295 Event Control Construct in always_comb

Configurable Parameter

Argument type: none;

Description

The rule checks if the event control construct @, is used in always_comb procedure. always_comb procedure represents combinational logics hence it should not contain any timing event.

Example

```
//////////////example : document.v////////////

module test(clk,data1,data2,y1,y2);
  input clk,data1,data2;
  output y1,y2;
  reg y1,y2;
  always_comb  begin
    y1 = data1;
    @(y1)
      y2 = data2;
  end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1295: Event construct @ is found in "always_comb" procedure.
```

Rule No 1296 Incomplete Assignment in always_comb

Configurable Parameter

Argument type: none;

Description

The rule checks whether the left-hand side variables in always_comb are assigned in every condition branch.

Example

```
//////////////example : document.v////////////

module test( en, a, b, a1, b1, a2, b2 );
  input en;
  output a, b, a1, b1, a2, b2;
  reg a, b, a1, b1, a2, b2;
  always_comb begin //warning on tmp.b and tmp.a here
    a = 0;
    if (en == 1 )
      b = 0;        //tmp.b is not assigned in all branches
    else a <= 1;    // tmp.a is not assigned in all branches
  end
  always_comb begin //warning on tmp.b1 here
    a1 = 0;         //tmp.a is fully assigned
    if (en == 1 )
      b1 = 0;       //tmp.b is not assigned in all branches
  end
  always_comb begin // No warning
    a2 = 0;         //tmp.a is fully assigned
    if (en == 1 )
      b2 = 0;       //tmp.b is fully assigned in all branches
    else b2 <= 1;
  end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1296: variable "test.b" and "a" is not completely assigned in always_comb block.
Violated 1296: variable "test.b1" is not completely assigned in always_comb block.
```

Rule No 1297 Real Value Compared in Case Item

Configurable Parameter

Argument type: none;

Description

The rule checks whether there are real comparisons in case item expressions. Real comparisons are typically not used in case items.

Example

```
//////////////example : document.v////////////

`define AA 1.1
module test;
 reg a,b,d;
 real c;
 parameter k=1.0;
 initial begin
   a=`AA;
   c=1.2;

   case (a)
     a              : b=0;
     1              : b=0;
     `AA             : b=1;  // Warning Here
     k              : b=1;  // Warning Here
     1.1,c           : b=1;  // Warning Here
     d              : b=1;
     default          : b=0;
   endcase

   case (b)
     `AA+1             : a=1;  // Warning Here
     k+1.0            : a=1;  // Warning Here
     c+1             : a=1;  // Warning Here
     `AA+((c+1)+(k+0.11))   : a=1;  // Warning Here
     k+`AA+((c+1)+(k+0.11)) : a=1;  // Warning Here
     default          : a=0;
   endcase
 end
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1297: Real expression is used in case item.
```

Rule No 1298 Report Snake Path

Configurable Parameter

Argument type: integer;
Argument description: specify the maximum length of snake path;
Default value: "10";
Argument Value: "2";

Description

Check to see there is any snake path. A violation will be reported when the snake path exceeds the specified argument value.

Example

```
//////////////example : document.v////////////

module dff (clk, i, o);
  input clk, i;
  output o;
  reg o;
  always @(posedge clk)
    o <= i;
endmodule

module comb (i1, i2, o);
  input i1, i2;
  output o;
  assign o = i1 & i2;
endmodule

module top (clk, i, o);
  input clk, i;
  output o;
  wire a1, a2, b1, b2, c1, c2, d;
  dff a_dff (.clk(clk), .i(i), .o(a1));
  comb a_comb (.i1(a1), .i2(a2), .o(b1));
  comb b_comb (.i1(b1), .i2(b2), .o(c1));
  comb c_comb (.i1(c1), .i2(c2), .o(d));
  dff c_dff (.clk(clk), .i(d), .o(o));
endmodule
```

Run Command :

```
iverilog-0.8 -tlint –S Document.v
```

Linting Tool reports:

```
Violated 1298: Snake path detected between register "top.a_dff.o :Reg" and register "top.c_dff.o: Reg".
```

Rule No 1299 Input Floating

Configurable Parameter

Argument type: none;

Description

Check to see there is any input floating of cell

Example

```
//////////////example : document.v////////////

module test (c);   //warning on PORT "c"
  output [7:0] c;
  test1 t1_i (c);
endmodule

module test1 (c);
  output [7:0] c;
  wire a;
  wire [7:0] c;
  not n1 (a, c[1]); // Warning as c is floating
  not n2 (c[2], a); // No Warning as c in not floating
endmodule
```

Run Command :

```
iverilog-0.8 -tlint Document.v
```

Linting Tool reports:

```
Violated 1299: Primary output pin "test.c" floating through wire "test.c".
Violated 1299: Primary output pin "test.t1_i.c" floating through wire "test.t1_i.c".
```

THANK YOU

---: @ THE END @ :---