Gowin 云源软件用户消息帮助文档

SUG937-1.0.4, 2023-12-08

版权所有 © 2023 广东高云半导体科技股份有限公司

GOWIN高云、Gowin、GowinSynthesis、云源以及高云均为广东高云半导体科技股份有限公司注册商标，本手册中提到的其他任何商标，其所有权利属其拥有者所有。未经本公司书面许可，任何单位和个人都不得擅自摘抄、复制、翻译本文档内容的部分或全部，并不得以任何形式传播。

免责声明

本文档并未授予任何知识产权的许可，并未以明示或暗示，或以禁止发言或其它方式授予任何知识产权许可。除高云半导体在其产品的销售条款和条件中声明的责任之外，高云半导体概不承担任何法律或非法律责任。高云半导体对高云半导体产品的销售和／或使用不作任何明示或暗示的担保，包括对产品的特定用途适用性、适销性或对任何专利权、版权或其它知识产权的侵权责任等，均不作担保。高云半导体对文档中包含的文字、图片及其它内容的准确性和完整性不承担任何法律或非法律责任，高云半导体保留修改文档中任何内容的权利，恕不另行通知。高云半导体不承诺对这些文档进行适时的更新。

版本信息

日期	版本	说明
2020/06/01	1.0	初始版本。
2020/08/06	1.0.1	● EX0311 ERROR 改成 EX0210 WARN； ● EX0206 WARN 代码更新； ● EX0310 ERROR 代码更新。
2023/07/14	1.0.2	新增 TA1125 NOTE。
2023/11/30	1.0.3	新增 TA1052 NOTE 和 TA1083 NOTE。
2023/12/08	1.0.4	更新 TA1125 NOTE。

目录

Gowin 云源软件用户消息帮助文档

概述

本手册主要描述高云®半导体云源®软件用户消息，旨在帮助用户快速处理软件使用过程中出现的 warning 信息及 error 信息。本手册包括 GowinSynthesis 用户消息和 Place & Route 用户消息，因软件版本升级，部分信息可能会略有差异，具体以用户软件版本的信息为准。

GowinSynthesis®用户消息

AG0100

WARN (AG0100): Find logical loop signal : <signal>

若设计中存在逻辑环路，综合工具通过上述警告列出环路所经过的<signal>，包括<signal>所在的行信息，可通过上述警告报出的行信息检查 RTL 设计修改逻辑环路以消除警告。逻辑环路的案例如下，设计中输出端口 out 驱动 out 本身，导致存在逻辑环路。

```
module test (in,out);
input in;
output out;
assign out = in & !out;
endmodule
```

Action

若要消除上述警告，则需要修改设计避免逻辑环路。

AG0101

WARN (AG0101): The netlist is not one directed acyclic graph including user instantiated primitives

若设计中存在逻辑环路，且逻辑环路中有用户实例化的逻辑原语，如 LUT、ALU 等，综合工具检测到上述环路给出警告并继续综合，举例如下，设计中 out1 和 out2 相互驱动对方导致逻辑环路，并且环路中存在用户实例化的

LUT3 ins1。

```
module test (a,b,out1,out2);
input a,b;
output out1,out2;
assign out1 = out2 & b;
LUT3 ins1(
    .I0(a),
    .I1(b),
    .I2(out1),
    .F(out2)
);
defparam ins1.INIT=8'hAB;
endmodule
```

Action

若要消除上述警告，则需要修改设计避免逻辑环路。

AG0200

ERROR (AG0200):The netlist is not one directed acyclic graph

同 AG0100 警告信息，若设计中存在逻辑环路，综合工具给出上述错误信息并停止综合。用户需要根据 AG0100 所报出的逻辑环路信息，修改用户设计将逻辑环路去掉并重新综合。

```
module test (in,out);
input in;
output out;
assign out = in & !out;
endmodule
```

Action

用户需要根据 AG0100 所报出的信息，修改用户设计避免逻辑环路并重新综合。

CK0013

ERROR (CK0013):<signal> is not connected to buf or iodelay.

基于芯片内部设计，某些逻辑单元间具有固定的连线，故这些单元<signal>的驱动源或者目的地会有固定的连线单元限制，如果连接到非固定的目标单元后，会报出以上错误。如下案例中，设计中例化了器件 oser4，oser4 的 Q0 驱动了逻辑操作 Q0&Q1，在器件布局中无法实现，故报出该错误信息。

```
    module OSER4_ins (Q0_test, D0, D1, D2, D3, TX0, TX1, PCLK, FCLK, RESET);
    input D0, D1, D2, D3;
    input TX0, TX1;
    input PCLK, FCLK, RESET;
    output Q0_test;
    wire Q0;
    wire Q1;
    OSER4 oser4(
   .Q0(Q0),
   .Q1(Q1),
   .D0(D0),
   .D1(D1),
   .D2(D2),
   .D3(D3),
   .TX0(TX0),
   .TX1(TX1),
   .PCLK(PCLK),
   .FCLK(FCLK),
   .RESET(RESET)
);
defparam oser4.GSREN = "false";
defparam oser4.LSREN = "true";
defparam oser4.HWL = "false";
defparam oser4.TXCLK_POL = 1'b0;
    assign Q0_test = Q0 & Q1;
    endmodule
```

Action

更改用户设计，使 Q0 直接驱动输出端口。

CK2027

ERROR(CK2027): The connection between Instance<inst1>and instance<inst2>is not correct!

基于芯片内部设计，某些逻辑单元<inst1>和<inst2>之间具有固定的连接，故这些单元的驱动源或者目的地会有固定的连接限制，如果连接到非固定的目标单元后，会报出以上错误。如下案例中，设计中例化了器件 DHCEN，DHCEN 的 CLKOUT 的输出为端口 CLKOUT，而根据芯片设计，DHCEN 的 CLKOUT 只能驱动 IOLOGIC/CLKDIV/DLL/PLL/DQS 等类型的时钟端口，故报出该错误信息。

```
    module DHCEN_ins (CLKOUT, CLKIN, CE);
    input CLKIN,CE;
    output CLKOUT;
```

```
DHCEN dhcen(
   .CLKOUT(CLKOUT),
   .CLKIN(CLKIN),
   .CE(CE)
);
endmodule
```

Action

将 DHCEN 的 CLKOUT 端口改为驱动正确的 instance。

```
module DHCEN_ins (Q0, CLKIN, CE);
input CLKIN,CE;
output Q0;
wire Q1;
wire D0;
wire D1;
wire D2;
wire D3;
wire TX0;
wire TX1;
wire PCLK;
wire RESET;
wire CLKOUT;
DHCEN dhcen(
   .CLKOUT(CLKOUT),
   .CLKIN(CLKIN),
   .CE(CE)
);
OSER4 oser4(
   .Q0(Q0),
   .Q1(Q1),
   .D0(D0),
   .D1(D1),
   .D2(D2),
   .D3(D3),
   .TX0(TX0),
   .TX1(TX1),
   .PCLK(PCLK),
   .FCLK(CLKOUT),
   .RESET(RESET)
);
defparam oser4.GSREN = "false";
defparam oser4.LSREN = "true";
defparam oser4.HWL = "false";
defparam oser4.TXCLK_POL = 1'b0;
```

```
endmodule
```

CV0003

WARN(CV0003):Output<port>has undriven bits, assigning undriven bits to 0,simulation mismatch possible

Output <port>悬空，将为其补上 GND 或高阻，综合结果的仿真行为可能会与 rtl 不同，综合工具给出上述警告信息并继续综合。如下案例中，output port o2 悬空。

```
module test(a,b,o1,o2);
input a,b;
output o1,o2; // o2 dangling
assign o1 = a & b;
endmodule
```

Action

如果某个 output port 连接内部信号，请将其赋值为 GND 或 VCC。

```
module test(a,b,o1,o2);
input a,b;
output o1,o2;
assign o1 = a & b;
assign o2 = 1'b0; // assign GND to dangling output port
endmodule
```

CV0005

ERROR(CV0005):Tran switch which all inputs are connected to inout port can not be converted

双向开关 tran 的所有 pin 不能全连 inout port，否则数据会对向冲突，综合工具给出上述错误信息并停止综合。如下案例中，tran 的两个 pin 都连接到了 inout port。

```
module test(io1,io2,control);
inout io1,io2;
input control;
tran t(io1,io2); //tran D0 and D1 all connect inout port
endmodule
```

Action

请将 tran 的两个 pin 分别连接不同类型的 port。

```
module test(o1,control);
```

```
output o1;
input control;
tran t(control,o1);//tran D0 connect input port, D1 connect output port
endmodule
```

CV0008

ERROR(CV0008):Convert tran switch <object> failed

双向开关 tran<object>的某个 pin 的连接使用不合法，导致无法将其综合，综合工具给出上述错误信息并停止综合。如下案例中，tran 的一个 pin 悬空。

```
module test(o1);
output o1;
wire control;
tran t(o1,control); // tran D1 dangling
endmodule
```

Action

请将悬空的 pin 连接 port 或内部信号。

```
module test(o1,control);
output o1;
input control;
tran t(control,o1); // connect tran D1 to output port
endmodule
```

CV0013

ERROR(CV0013):Tri-state signal <signal> connected ERROR, it should only be connected to PAD.

三态 Buf 的信号<signal>只能连接 PAD，不能连接其他类型的信号。如下的用户例化 IOBUFgwBuf 的三态信号 io 虽连接 PAD，但同时被 GND(2'b00)驱动，报出该错误信息。

```
module InterFace(
    inout io,
    input  in, io_en,
    output lrck
);
IOBUF gwBuf(
.I(in),
.OEN(io_en),
.IO(io),
```

```
    .O(lrck)
    );
    assign io = 2'b00;  //assign GND to io
    endmodule
```

Action

移除信号 io 的驱动源 2’b00。

```
    module InterFace(
        inout io,
        input  in, io_en,
        output lrck
    );
    IOBUF gwBuf(
    .I(in),
    .OEN(io_en),
    .IO(io),
    .O(lrck)
    );
    endmodule
```

CV0014

ERROR(CV0014):Not Support MOS switch<signal>synthesis

不支持实例化的 cmos/rcmos 的综合，综合工具给出上述错误信息并停止综合。如下案例中，实例化一个 cmos。

```
    module test(in,control1,control2,o);
    input in,control1,control2;
    output o;
    cmos c(o,in,control1,control2);
    endmodule
```

Action

请删除 cmos/rcmos 的实例化，并用其他逻辑门代替其功能。

DI0002

WARN(DI0002):Asynchronous register<asynReg >initial values do not match with the Gowin library, simulation mismatch possible

根据芯片设计，异步置位触发器<asynReg>的初始值仅可设置为 1，异步复位触发器<asynReg>的初始值仅可设置为 0，否则综合工具将给出上述

警告信息。如下案例中， Register o 是一个异步清零的 D 触发器，其初始值仅可为 0，但在 RTL 中被设置为 1。

```
    module test(clk,d,clear,o);
    input clk,d,clear;
    output o;
    reg o = 1'b1;
    always @(posedge clk or posedge clear) // async register
    if(clear)
    o = 0; // register output 0 if clear, this register will be synthesized
to DFFC
    else
    o = d;
    endmodule
```

Action

对<asynReg>，可以不设置初始值，或初始值与<asynReg>的 clear/preset 信号的复位/置位结果保持一致。

```
    module test(clk,d,clear,o);
    input clk,d,clear;
    output o;
    reg o = 1'b0;
    always @(posedge clk or posedge clear) // async register
    if(clear)
    o = 0; // register output 0 if clear, this register will be synthesized
to DFFC
    else
    o = d;
    endmodule
```

EX0200

WARN (EX0200): Property <prop>set invalid for <object>

当设计中<object>设置了属性约束<prop>，但是约束属性值设置不合理时，综合工具会给出上述警告并且采用默认初始值。在如下的测试案例中，属性约束 syn_ramstyle 赋值为空，将会报出上述警告信息。

```
    module normal1(data_out, data_in, addr, clk, wre,rst);
    output [1:0]data_out;
    input [1:0]data_in;
    input [6:0]addr;
    input clk,wre,rst;
    reg [1:0] mem [127:0] /* synthesis syn_ramstyle = "" */;
    reg [1:0] data_out;
```

```
always@(posedge clk or posedge rst)
 if(rst ==1)
        data_out <= 0;
 else
    if(wre == 0)
        data_out <= mem[addr];
 always @(posedge clk)
    if (wre) mem[addr] <= data_in;
endmodule
```

Action

确保设计中使用的属性约束赋值正确，如下更正的测试案例中，syn_ramstyle 赋值 registers。

```
module normal1(data_out, data_in, addr, clk, wre,rst);
output [1:0]data_out;
input [1:0]data_in;
input [6:0]addr;
input clk,wre,rst;
reg [1:0] mem [127:0] /* synthesis syn_ramstyle = "registers" */;
reg [1:0] data_out;
always@(posedge clk or posedge rst)
 if(rst ==1)
        data_out <= 0;
 else
    if(wre == 0)
        data_out <= mem[addr];
 always @(posedge clk)
    if (wre) mem[addr] <= data_in;
endmodule
```

EX0201

WARN (EX0201): Missing INIT parameter on <object> and using default value

当设计中实例化<object>，但<object>未设置初始值，综合工具给出上述警告并设置默认初始值。在如下的测试案例中，设计中实例化 LUT3 ins1，但 ins1 未设置 INIT 初始值。

```
module test(a,b,c,out);
input a,b,c;
output out;
LUT3 ins1(
    .I0(a),
```

```
        .I1(b),
        .I2(c),
        .F(out)
    );
    /*lack defparam of LUT3*/
    endmodule
```

Action

若要消除上述警告，则需要给实例化的<object>赋初始值，如下所示。

```
    module test(a,b,c,out);
    input a,b,c;
    output out;
    LUT3 ins1(
        .I0(a),
        .I1(b),
        .I2(c),
        .F(out)
    );
    defparam ins1.INIT=8'hEF;
    endmodule
```

EX0203

WARN (EX0203): Top module <modu> has no ports

若设计中顶层模块<modu>没有任何端口，则综合工具给出上述警告，示例如下，模块 test 没有任何端口信息。

```
    module test();
    wire a,b,out;
    assign a = 1'b0;
    assign b = 1'b1;
    assign out = a ^ b;
    endmodule
```

Action

若要消除上述警告信息，顶层模块<modu>需要设置至少一个输入或输出端口。

EX0205

WARN (EX0205): Instance <inst> 's parameter <para> value invalid, replaced by default value <para>

若设计中例化的高云原语<inst>的<para>数值错误，则综合工具给出上

述警告信息，并将<inst>的<para>替换成默认值。示例如下，设计中 LUT4 的<para>数值错误。

```
module unitest ( i,out);
input [3:0] i;
output out;
LUT4 lut4_0 (
    .I0(i[0]),
    .I1(i[1]),
    .I2(i[2]),
    .I3(i[3]),
    .F(out)
);
defparam lut4_0.INIT = "wabajd";///< right: 16'h0000 to 16'hFFFF
endmodule
```

Action

若要消除上述警告信息，需要参考 SUG283，Gowin 原语用户指南将例化的高云原语的<para>设置在合理范围内。

```
module unitest ( i,out);
input [3:0] i;
output out;
LUT4 lut4_0 (
    .I0(i[0]),
    .I1(i[1]),
    .I2(i[2]),
    .I3(i[3]),
    .F(out)
);
defparam lut4_0.INIT = 16'h1234;///< right: 16'h0000 to 16'hFFFF
endmodule
```

EX0206

WARN (EX0206): Instance <inst> 's parameter <para> value invalid

当前指定器件与例化的 rPLL 等高云原语<inst>的<para>中指定的器件不一致时，综合工具给出上述警告。示例如下，指定的器件信息与 rPLL 器件 GW1N-4B 不一致时，会报出此警告信息。

```
module test(i,out);
input [35:0]i;
output [4:0]out;
rPLL rpll (
```

```
        .CLKIN(i[0]),
        .CLKFB(i[1]),
        .FBDSEL(i[7:2]),
        .IDSEL(i[13:8]),
        .ODSEL(i[19:14]),
        .DUTYDA(i[23:20]),
        .PSDA(i[27:24]),
        .FDLY(i[31:28]),
        .RESET(i[32]),
        .RESET_P(i[33]),
        .CLKOUT(out[0]),
        .CLKOUTP(out[1]),
        .CLKOUTD(out[2]),
        .CLKOUTD3(out[3]),
        .LOCK(out[4])
    );
    defparam rpll.DEVICE = "GW1N-4B";
    endmodule
```

Action

若要消除上述警告，需要将指定器件信息与例化的高云原语<para>中的器件信息保持一致。

EX0210

WARN (EX0210): Invalid input frequency <freq> to instance <inst>, suitable range is from <num1>MHz to <num2>MHz

若综合工具报出上述警告信息，则说明例化的高云原语器件<inst>的<freq>不在合法的<num1>到<num2>范围内，请根据报错的<inst>名称及行信息修改相关设计。例化的高云原语<inst>未设置的<freq>按默认<freq>的数值进行检查。

```
    module test(i,out);
    input [35:0]i;
    output [4:0]out;
    rPLL rpll (
        .CLKIN(i[0]),
        .CLKFB(i[1]),
        .FBDSEL(i[7:2]),
        .IDSEL(i[13:8]),
        .ODSEL(i[19:14]),
        .DUTYDA(i[23:20]),
        .PSDA(i[27:24]),
        .FDLY(i[31:28]),
```

```
        .RESET(i[32]),
        .RESET_P(i[33]),
        .CLKOUT(out[0]),
        .CLKOUTP(out[1]),
        .CLKOUTD(out[2]),
        .CLKOUTD3(out[3]),
        .LOCK(out[4])
    );
    defparam rpll.FCLKIN = "100.0";
    defparam rpll.DYN_IDIV_SEL = "false";
    defparam rpll.IDIV_SEL = 0;
    defparam rpll.DYN_FBDIV_SEL = "false";
    defparam rpll.FBDIV_SEL = 0;
    defparam rpll.DYN_ODIV_SEL = "false";
    defparam rpll.ODIV_SEL = 32;
    defparam rpll.PSDA_SEL = "0000";
    defparam rpll.DYN_DA_EN = "false";
    defparam rpll.DUTYDA_SEL = "1000";
    defparam rpll.CLKOUT_FT_DIR = 1'b1;
    defparam rpll.CLKOUTP_FT_DIR = 1'b1;
    defparam rpll.CLKOUT_DLY_STEP = 0;
    defparam rpll.CLKOUTP_DLY_STEP = 0;
    defparam rpll.CLKFB_SEL = "internal";
    defparam rpll.CLKOUT_BYPASS = "false";
    defparam rpll.CLKOUTP_BYPASS = "false";
    defparam rpll.CLKOUTD_BYPASS = "false";
    defparam rpll.DYN_SDIV_SEL = 2;
    defparam rpll.CLKOUTD_SRC = "CLKOUT";
    defparam rpll.CLKOUTD3_SRC = "CLKOUT";
    defparam rpll.DEVICE = "GW1N-4B";
    endmodule
```

Action

若要消除上述错误，需要参考报错信息及 SUG283，Gowin 原语用户指南调整<freq>值的大小，或调整指定的器件信息。

EX0302

ERROR (EX0302): No valid top module found

若设计文件中没有任何 module，设计文件为空，综合工具会报出上述错误信息。不允许综合空的设计文件。

EX0308

ERROR(EX0308):GowinSynthesis can not find file \"primitive.xml\". Please reinstall the product

若安装目录 IDE\bin 下的综合配置文件 primitive.xml 被移动，综合工具给出上述错误信息。请尝试将 primitive.xml 移动回原有位置，或重新安装软件。

EX0309

ERROR (EX0309): Net <object> has multiple drivers

若设计中存在一条信号<object>具有多个非三态的驱动源，综合工具给出上述错误信息。可根据错误信息提示的信号名和文件行信息进行查找及修改。示例如下，输出端口 out 同时被 rotate_1 和 rotate_2 驱动。

```
module rotate (q1, data, sel1) ;
output [7:0] q1;
input [7:0] data;
input sel1;
endmodule

module top (out, ci, data1, data2);
output [7:0] out;
input [7:0] data1, data2;
input ci;
rotate rotate_1 (out, data1, ci);
rotate rotate_2 (out, data2, ci);
endmodule
```

Action

若要消除上述错误，需要修改设计使一条信号只有一个驱动源，如下所示。

```
module rotate (q1, data, sel1) ;
output [7:0] q1;
input [7:0] data;
input sel1;
endmodule

module top (out, ci, data1);
output [7:0] out;
input [7:0] data1;
input ci;
rotate rotate_1 (out, data1, ci);
endmodule
```

EX0310

ERROR (EX0310): Invalid parameterized value <paraValue>(<para>) specified for instance <inst>

若综合工具报出上述错误信息，则说明例化的高云原语<inst>的<para>的数值<paraValue>不正确，请根据报错的名称及行信息修改相关设计。示例如下，rPLL 的参数 FCLKIN 的数值超出合理范围。

```
module test(i,out);
input [35:0]i;
output [4:0]out;
rPLL rpll (
    .CLKIN(i[0]),
    .CLKFB(i[1]),
    .FBDSEL(i[7:2]),
    .IDSEL(i[13:8]),
    .ODSEL(i[19:14]),
    .DUTYDA(i[23:20]),
    .PSDA(i[27:24]),
    .FDLY(i[31:28]),
    .RESET(i[32]),
    .RESET_P(i[33]),
    .CLKOUT(out[0]),
    .CLKOUTP(out[1]),
    .CLKOUTD(out[2]),
    .CLKOUTD3(out[3]),
    .LOCK(out[4])
);
defparam rpll.DEVICE = "GW1N-4B";
defparam rpll.FCLKIN = "600.0";
endmodule
```

Action

若要消除上述错误，请参考 SUG283，Gowin 原语用户指南确保<inst>的<para>对应的<paraValue>合理。

EX2526

WARN (EX2526): Entry size <width> at <initvalue>:<initWidth> does not match memory width <memWidth>

若设计中使用$readmemh 语句，但对应文件数据宽度<width>不匹配<memWidth>时，综合工具会给出上述警告信息。如下案例中 mem 宽度是 8，如果对应<initvalue>文件数据宽度<initWidth>不匹配，就会报出此警告信息。

```
module test(in,clk,addr_in,addr_out,out);
input in,clk;
input addr_in;
input addr_out;
output out;
reg mem[7:0];
 always @ (posedge clk)
     mem[addr_in] <= in;
 assign out = mem[addr_out];
initial begin
    $readmemh("initvalue", mem);
  end
endmodule
```

Action

若要消除上述警告，需要$readmemh()语句指定的数据文件和数组宽度匹配。

EX2598

WARN (EX2598): <design> might have mixed concurrent and procedural assignment

若设计<design>中可能同时存在阻塞赋值和非阻塞赋值的情况，综合工具会给出上述警告信息。如下案例寄存器 d_reg 在不同条件下同时存在阻塞赋值和非阻塞赋值的情况。

```
module gene_if(t0,t1,t2,d,clk,t);
input t0,t1,t2,clk,t;
output d;
reg d_reg;
localparam S=6;
generate
    if(S<7)
 assign d_reg=t0|t1|t2;
```

```
    else
assign d_reg=t0&t1&t2;
endgenerate
generate
    if(S>7)
        always @(posedge clk)
        d_reg<=t;
        else
        always @(posedge clk)
        d_reg<=t0&t1&t2;
endgenerate
assign d=d_reg;
endmodule
```

Action

若要消除上述警告，需要将重复赋值的情况删除，如下所示。

```
module gene_if(t0,t1,t2,d,clk,t);
input t0,t1,t2,clk,t;
output d;
reg d_reg;
localparam S=6;
generate
    if(S>7)
        always @(posedge clk)
        d_reg<=t;
        else
        always @(posedge clk)
        d_reg<=t0&t1&t2;
endgenerate
assign d=d_reg;
```

```
endmodule
```

EX2629

WARN (EX2629): Delay control is not supported for synthesis

延时语句为不可综合语句，如果设计文件包含延时如#10 语句，综合工具会给出上述警告信息，所有的延时会被忽略。

```
module top (in0,in1,clk,out);
input in0,in1;
input clk;
output reg out;
always @( posedge clk)
    begin
        out <= #10 in0&in1;
    end
endmodule
```

Action

若要消除上述警告，需要移除设计文件中的延时控制，如下所示。

```
module top (in0,in1,clk,out);
input in0,in1;
input clk;
output reg out;
always @( posedge clk)
    begin
        out <= in0&in1;
    end
endmodule
```

EX2635

WARN (EX2635): Generate block is allowed only inside loop and conditional generate in SystemVerilog mode

若设计中使用 generate 语句循环生成模块，综合工具会给出上述警告信息。如下案例在 for 循环中生成 test 模块，此用法仅在 SystemVerilog 支持。

```
module top(in,out);

input [1:0]in;

output [1:0]out;

generate
```

```
begin
   genvar i;
   for (i=0;i<2;i=i+1)
   begin : reg_loop1
test test1(in[i],out[i]);
   end
end
endgenerate
endmodule
module test(in,out);
input in;
output out;
assign out = !in;
endmodule
```

Action

上述警告是使用此语法的正常现象，注意此用法仅在 SystemVerilog 支持。

EX2656

ERROR (EX2656): SystemVerilog keyword <word> used in incorrect context

若设计文件定义变量与关键字<word>重名，综合工具会给出上述错误信息。null 是 system verilog 中的一个关键字，不能用于变量名。

```
module top (in0,in1,out);
input in0,in1;
output out;
wire null;
assign  out= in0&in1;
endmodule
```

Action

若要消除上述错误，需要注意避免变量名与关键字重名，如下所示。

```
module top (in0,in1,out);
input in0,in1;
```

```
output out;
assign  out= in0&in1;
endmodule
```

EX2664

WARN (EX2664): Variable <vari> may be used before assigned in always_comb or always @* block : might cause synthesis - simulation differences

若设计中存在 always 语句敏感列表内信号<vari>在 always 内变化的情况，综合工具会给出上述警告信息。如下案例 tmp 在敏感列表内，但不断变化。

```
module top(in,sel,out);

input in,sel;

output reg out;

reg tmp;

always@(*)

    begin

        if(sel)

            tmp <= in;

        else

            tmp <= !tmp;

    end

assign out= tmp;

endmodule
```

Action

若要消除上述警告，需要添加时钟信号，将 tmp 信号移出敏感列表。如下所示。

```
module top(in,sel,out,clk);

input in,sel,clk;

output reg out;

reg tmp;

always@(posedge clk)
```

```
    begin
        if(sel)
            tmp <= in;
        else
            tmp <= !tmp;
    end
assign out= tmp;
endmodule
```

EX2565

WARN (EX2565): Port <port> is not connected on this instance

若设计中存在定义一个端口<port>，但没有使用的情况，综合工具会给出上述警告信息。如下案例 test 模块中 clk 端口没有使用。

```
module top (in0,in1,out,out1);
input in0,in1;
output out,out1;
assign out = in0 & !in1;
test test1(
        .data(in0),
        .out(out1)
);
endmodule
module test (data,out,clk);
input data,clk;
output out;
assign out = !data;
endmodule
```

Action

若要消除上述警告，需要移除无效端口，如下所示。

```
module top (in0,in1,out,out1);
```

```
input in0,in1;
output out,out1;
assign out = in0 & !in1;
test test1(
        .data(in0),
        .out(out1)
);
endmodule
module test (data,out);
input data;
output out;
assign out = !data;
endmodule
```

EX2666

WARN (EX2666): Unsupported use of clock signal <signal>, clock used as data

若设计中使用同一个信号<signal>作为时钟和数据，综合工具会给出上述警告信息，如下案例 clk 信号同时作为时钟和数据。

```
module top (clk,out);
input clk;
output reg out;
always@(posedge clk)
    out <= clk;
endmodule
```

Action

若要消除上述警告，需要将输入数据和时钟分离，如下所示。

```
module top (in,clk,out);
input in,clk;
output reg out;
```

```
always@(posedge clk)

    out <= in;

endmodule
```

EX2830

WARN (EX2830): Data object <object> is already declared

设计中对同一个变量<object>重复定义时，综合工具会给出上述警告信息，如下案例重复定义 2 次 wire 赋值，需要删除其中的一个定义。

```
module top (in,out);
input in;
output out;
wire out;
wire out;
assign out = !in;
endmodule
```

Action

若要消除上述警告，需要将重复定义删除，如下所示。

```
module top (in,out);
input in;
output out;
wire out;
assign out = !in;
endmodule
```

EX2855

WARN (EX2855): Result of this <oper> operation does not fit in <width> bits

若设计中使用运算操作符<oper>，但<oper>的结果位宽超过赋值位宽时，综合工具会给出上述警告信息。如下案例幂运算结果位宽超过 out 端口。

```
module top(in,out);

input in;

output [1:0]out;

assign out = 6'd2 ** (16'h77)+in;

endmodule
```

Action

若要消除上述警告，需要赋值操作两端位宽相同，如下所示。

```
module top(in,out);

input in;

output [1:0]out;

assign out = 2'b01 ** (2'b10)+in;

endmodule
```

EX2932

WARN (EX2932): Unknown system task <task> ignored for synthesis

若设计中存在未知的系统任务<task>，综合工具会给出上述警告信息。如下案例仿真命令$fsdbDumpMDA 不被综合识别，综合时会忽略未知系统任务。

```
module test(in,out);

input in;

output out;

reg mem;

assign out = in;

initial begin

$fsdbDumpMDA(mem);

end

endmodule
```

Action

若要消除上述警告，需要删除未知系统任务，如下所示。

```
module test(in,out);

input in;

output out;

reg mem;

assign out = in;

endmodule
```

EX2947

WARN (EX2947): Input port <port> remains unconnected for this instance

若模块输入端口<port>实例化时，未连接相应信号，综合工具会给出上述警告信息。如下案例实例化 sub，但没有将 in1 连接。

```
module top (top_in0,top_in1,top_out);
input top_in0,top_in1;
output top_out;
sub sub1(
.in0(top_in0),
.in1(),
.out(top_out)
);
endmodule
module sub (in0,in1,out);
input in0,in1;
output out;
assign out = in0 & !in1;
endmodule
```

Action

若要消除上述警告，可以对悬空端口连接，如下所示。

```
module top (top_in0,top_in1,top_out);
input top_in0,top_in1;
output top_out;
sub sub1(
.in0(top_in0),
.in1(top_in1),
.out(top_out)
);
endmodule
module sub (in0,in1,out);
input in0,in1;
output out;
assign out = in0 & !in1;
endmodule
```

EX2981

WARN (EX2981): Net <objec> is driven by multiple input ports

若设计中存在输入端口重复给同一个变量<object>赋值的情况，综合工具会给出上述警告信息。如下案例 tmp 被重复赋值 2 次。

```
module test(in,out);

input [3:0]in;

output [3:0]out;

wire [3:0]tmp;

assign tmp[3:0] = in;

assign tmp[1:0] = in;

assign out = tmp;

endmodule
```

Action

若要消除上述警告，需要将重复赋值删除，如下所示。

```
module test(in,out);

input [3:0]in;

output [3:0]out;

wire [3:0]tmp;

assign tmp[3:0] = in;

assign out = tmp;

endmodule
```

EX2987

WARN (EX2987): Input port <port> is not connected on this instance

若模块端口<port>实例化时，未连接相应信号，综合工具会给出上述警告信息。如下案例实例化 test 时，没有将 in 端口连接，in 端口在 test 模块中悬空。

```
module top (in0,out1);
input in0;
output out1;
test test1(
.in1(in0),
.in(),
.out(out1)
```

```
);
endmodule
module test (in1,in,out);
input in1;
input in;
output out;
assign out = !in1;
endmodule
```

Action

若要消除上述警告，需要将无用的悬空端口删除，如下所示。

```
module top (in0,out1);
input in0;
output out1;
test test1(
.in1(in0),
.out(out1)
);
endmodule
module test (in1,out);
input in1;
output out;
assign out = !in1;
endmodule
```

EX2997

WARN (EX2997): Net <net> is already driven by input port <port>

若设计中存在给<net>的输入端口<port>赋值的情况，综合工具会给出上述警告信息。如下案例输入端口 d 被其他输入端口驱动。

```
module test(b,c,d,f);
input b,c,d;
output f;
assign d = c&b;
assign f = b&d;
endmodule
```

Action

若要消除上述警告，需要将此情况删除，如下所示。

```
module test(b,d,f);
input b,d;
output f;
```

```
assign f = b&d;
endmodule
```

EX2998

WARN (EX2998): Net <object> does not have a driver

若设计中存在定义一个连线或寄存器类型变量<object>，但没有驱动的情况，综合工具会给出上述警告信息。如下案例连线 a 被定义和使用，但没有驱动。

```
module top (in0,in1,out);
input in0,in1;
output out;
wire a;
assign out = in0&in1|a;
endmodule
```

Action

若要消除上述警告，需要将连线 a 删除，或添加相应的连接关系，如下所示。

```
module top (in0,in1,out);
input in0,in1;
output out;
assign out = in0&in1;
endmodule
```

EX2999

ERROR (EX2999): Another driver from here

若设计中存在多个输入同时驱动同一个输出的情况，综合工具会给出上述错误信息，分别指出重复赋值的位置，如下案例 out 端口被重复赋值，需要删除其中一个。

```
module test (in0,in1,out);
input in0,in1;
output out;
assign out = in0 & !in1;
assign out = in0 & in1;
endmodule
```

Action

若要消除上述错误，需要将重复赋值删除，如下所示。

```
module test (in0,in1,out);
input in0,in1;
output out;
assign out = in0 & !in1;
endmodule
```

EX3000

ERROR (EX3000): Net <object> is constantly driven from multiple places

若设计中存在多个输入同时驱动同一个输出<object>的情况，综合工具会给出上述错误信息，指出重复赋值的位置，如下案例 out 端口被重复赋值，需要删除其中一个。

```
module test (in0,in1,out);
input in0,in1;
output out;
assign out = in0 & !in1;
assign out = in0 & in1;
endmodule
```

Action

若要消除上述错误需要将重复赋值删除，如下所示。

```
module test (in0,in1,out);
input in0,in1;
output out;
assign out = in0 & !in1;
endmodule
```

EX3041

WARN (EX3041): <object> shift count >= width of value

若设计中存在移位一个变量<object>，但移位数大于变量位宽的情况，综合工具会给出上述警告信息。如下案例 in1 位宽为 1，左移 2 位。此情况值恒为 0，是无效的。

```
module top (in0,in1,out);

input in0,in1;

output reg out;

assign out = in0&(in1 << 2);

endmodule
```

Action

若要消除上述警告，需要删除无效移位操作。

EX3044

WARN (EX3044): Overwriting previous value of parameter <para>

若设计中重复给同一个<para>赋值，综合工具会给出上述警告信息。如下案例 LUT2 的 INIT 被赋值 2 次。

```
module top (in0,in1,out);
input in0,in1;
output reg out;
LUT2 lut2(
.I0(in0),
.I1(in1),
.F(out));
defparam lut2.INIT = 4'h4;
defparam lut2.INIT = 4'h6;
endmodule
```

Action

若要消除上述警告，需要将重复赋值删除，如下所示。

```
module top (in0,in1,out);
input in0,in1;
output reg out;
LUT2 lut2(
.I0(in0),
.I1(in1),
.F(out));
defparam lut2.INIT = 4'h6;
endmodule
```

EX3073

WARN (EX3073): Port <port> remains unconnected for this instance

模块例化时，若模块定义的端口<port>在模块例化端口列表中不存在，综合工具会给出上述警告信息。如下案例中实例化 sub 时未赋值 out1 端口。

```
module top (top_in,top_out);
input top_in;
output top_out;
sub sub1(
.in(top_in),
.out0(top_out)
);
endmodule
module sub (in,out0,out1);
input in;
output out0;
output out1;
assign out0 = !in;
endmodule
```

Action

若要消除上述警告，需要删除 out1 端口或在例化时添加 out1 的连接关系，如下所示。

```
module top (top_in,top_out);
input top_in;
output top_out;
sub sub1(
.in(top_in),
.out0(top_out)
);
endmodule
module sub (in,out0);
input in;
output out0;
assign out0 = !in;
endmodule
```

EX3359

ERROR (EX3359): Null as source expression is not allowed here

若设计文件将 null 赋值给某个信号时，综合工具会给出上述错误信息，null 是 system verilog 中的一个关键字。

```
module top (in,out);
input in;
output out;
assign  out= null;
endmodule
```

Action

若要消除上述错误，需要避免错误使用关键字。

EX3413

ERROR (EX3413): Second argument of '$<object> must be a memory

若设计中错误使用$<object>语法，综合工具会给出上述错误信息。如下案例$readmemh 语法的第二个参数 mem 应该是一个二维数组。

```
module test(in,clk,addr_in,addr_out,out);
input in,clk;
input addr_in;
input addr_out;
output out;
reg [7:0]mem;
 always @ (posedge clk)
     mem[addr_in] <= in;
 assign out = mem[addr_out];
initial begin
    $readmemh("initvalue", mem);
  end
endmodule
```

Action

若要消除上述错误，需要将 mem 定义为二维数组，如下所示。

```
module test(in,clk,addr_in,addr_out,out);
input in,clk;
```

```
input addr_in;
input addr_out;
output out;
reg [7:0]mem[7:0];
 always @ (posedge clk)
     mem[addr_in] <= in;
 assign out = mem[addr_out];
initial begin
    $readmemh("initvalue", mem);
  end
endmodule
```

EX3483

ERROR (EX3483): Cannot open Verilog file <file>

若指定的设计文件<file>不存在或没有权限打开时，综合工具会给出上述错误信息，请检查文件是否存在并检查文件权限。

EX3514

ERROR (EX3514): Module <modu> in library <lib> is not yet analyzed

若在库<lib>中不存在指定的<modu>时，综合工具会给出上述错误信息，请核对指定的<modu>名称是否正确。

EX3534

ERROR (EX3534):Assignment under multiple single edges is not supported for synthesis

若设计中 always 语句中的一个敏感信号在一个敏感信号列表中既有上升沿触发又有下降沿触发，综合工具会给出上述错误信息。如下案例 clk 信号同时包含上升沿和下降沿触发，这种情况没有对应类型的高云原语，无法进行逻辑映射。

```
module top (in,out,clk,clear);
input in,clk,clear;
output reg out;
always @(posedge clk or negedge clk)
```

```
        if(clear)
            begin
                out <= 1'b0;
            end
        else
            begin
                out <= in;
            end
    endmodule
```

Action

若要消除上述错误需要将 clk 信号的上升沿触发或者下降沿触发删除其中一个，如下所示，只保留上升沿触发。

```
    module top (in,out,clk,clear);
    input in,clk,clear;
    output reg out;
    always @(posedge clk)
        if(clear)
            begin
                out <= 1'b0;
            end
        else
            begin
                out <= in;
            end
    endmodule
```

EX3589

ERROR (EX3589): Keyword <object> is not allowed here in this mode of Verilog

若设计文件错误使用关键字<object>时，综合工具会给出上述错误信息。

EX3628

WARN (EX3628): Redeclaration of ansi port <port> is not allowed

若设计中存在将输出端口<port>作为赋值右值的情况，综合工具会给出上述警告信息。如下案例 ClkOut 在 always 语句中作为右值进行取反操作。

```
    module top(

        input ClkIn,

        input rst,
```

```
        output ClkOut
        );
        reg ClkOut;
        always@(posedge ClkIn)
        begin
        if(rst) ClkOut = 1'b0;
    else ClkOut = ~ClkOut;
        end

    endmodule
```

Action

若要消除上述警告，可以定义一个中间寄存器，最后再赋值给输出端口。如下所示。

```
    module top(
        input ClkIn,
        input rst,
        output ClkOut
        );
        reg tmp;
        always@(posedge ClkIn)
        begin
        if(rst) tmp = 1'b0;
        else tmp = ~tmp;
        end
        assign ClkOut = tmp;
    endmodule
```

EX3638

WARN (EX3638) : <object> is already implicitly declared on line <lineInfo>

若设计在<lineInfo>中使用隐式声明连线<object>，但又在使用后显式声明，综合工具会给出此条警告信息。如下案例连线 tmp 在例化模块 aa 时隐式声明并使用，但在之后显式声明。

```
module top (in0,in1,out);
input in0,in1;
output  out;
aa ins(in0,in1,tmp);
wire tmp;
assign out = tmp;
endmodule
module aa(in0,in1,out);
input in0,in1;
output out;
assign out=in0|| in1;
endmodule
```

Action

若要消除上述警告，可以将连线声明放在使用之前，或直接采用隐式声明，删除显式声明。如下所示。

```
module top (in0,in1,out);
input in0,in1;
output  out;
wire tmp;
aa ins(in0,in1,tmp);
assign out = tmp;
endmodule
module aa(in0,in1,out);
input in0,in1;
output out;
assign out=in0|| in1;
endmodule
```

EX3670

WARN (EX3670):Actual bit length <actlen> differs from formal bit length <forlen>for port <port>

对设计中的模块<port>例化时，若端口例化位宽<actlen>与定义时的位宽<forlen>不匹配时，综合工具会给出上述警告信息。如下案例实例化 test 时 in 端口和输入 top_in 的宽度不同，out 端口和输入 top_out 的宽度不同。

```
module top (top_in,top_out);
input top_in;
output top_out;
test test1(
.in(top_in),
.out(top_out)
);
endmodule
module test (in,out);
input [2:0]in;
output [1:0]out;
assign out[0] = in[0];
assign out[1] = in[1] & !in[2];
endmodule
```

Action

若要消除上述警告，需要保持相应端口位宽相同，如下所示。

```
module top (top_in,top_out);
input [2:0]top_in;
output [1:0]top_out;
test test1(
.in(top_in),
.out(top_out)
);
endmodule
module test (in,out);
input [2:0]in;
output [1:0]out;
assign out[0] = in[0];
assign out[1] = in[1] & !in[2];
endmodule
```

EX3671

WARN (EX3671): Second declaration of <object> ignored

设计中对同一个变量<object>重复定义时，综合工具会给出上述警告信息，如下案例对 out 重复定义 2 次。

```
module top (in,out);
input in;
output out;
wireout;
wireout;
assign out = !in;
endmodule
```

Action

若要消除上述警告，需要将重复定义删除，如下所示。

```
module top (in,out);
input in;
output out;
wireout;
assign out = !in;
endmodule
```

EX3680

WARN (EX3680): Concatenation with unsized literal,will interpret as 32 bits

若设计中使用一个未定义位宽的变量组合赋值，综合工具会给出此条警告信息。如下案例'b0 未定义位宽，将默认拼接成 32 位宽变量，可能会造成位宽不匹配或原有位数缺失。

```
module test (in,out);

input in;

output [15:0]out;

assign out = {'b0,in};

endmodule
```

Action

若要消除上述警告，组合位宽中的参数必须是固定位宽，如下所示。

```
module test (in,out);

input in;

output [15:0]out;

assign out = {15'b0,in};
```

```
endmodule
```

EX3682

WARN (EX3682): Variable <vari> might have multiple concurrent drivers

若设计中 1 个输出端口<vari>可能存在多个驱动时，综合工具会给出上述警告信息。如下案例 out 端口可能存在多个驱动。

```
module top(in,sel,out);
input in,sel;
output reg out;
reg tmp;
always@(*)
    begin
        if(sel)
            tmp <= in;
        else
            out <= !tmp;
    end
assign out= tmp;
endmodule
```

Action

若要消除上述警告，需要避免多个输入驱动一个端口的情况。

EX3705

WARN (EX3705): Macro <object> redefined

若设计中多次使用 define 语句定义同一个参数<object>时，综合工具会给出上述警告信息。在后方的 define 语句会替换前方的定义。如下案例 INIT 重复定义了不同值，会导致 out0 端口和 out1 端口的值不同。

```
`define INIT 1'b0
module test (in,out0,out1);
input in;
output out0,out1;
assign out0 = !in|`INIT;
```

```
`define INIT 1'b1
assign out1 = !in|`INIT;
endmodule
```

Action

若要消除上述警告，建议将其定义成 2 个不同 define，避免歧义，或者使用 if define 语句。

```
`define INIT0 1'b0
`define INIT1 1'b1
module test (in,out0,out1);
input in;
output out0,out1;
assign out0 = !in|`INIT0;
assign out1 = !in|`INIT1;
endmodule
```

EX3706

WARN (EX3706): Empty port in <modu> declaration

若设计中模块<modu>声明不正确，综合工具会给出上述警告信息，如下案例声明位置多加了“,”。

```
module test(in,out,);

input in;

output out;

assign out = !in;

endmodule
```

Action

若要消除上述警告，需要删除多余的“,”，如下所示。

```
module test(in,out);

input in;

output out;

assign out = !in;

endmodule
```

EX3735

ERROR (EX3735): Port <port> is already connected

若设计中实例化 instance 时重复给一个端口<port>赋值，综合工具会给出上述错误信息，如下案例实例化 test 时将 2 个不同的值赋给 in 端口。

```
module test (in,out);
input in;
output out;
assign out = !in;
endmodule
module top (in0,in1,out0);
input in0,in1;
output out0;
test test1(
.in(in0),
.in(in1),
.out(out0)
);
endmodule
```

Action

若要消除上述错误，需要将重复的端口赋值删除一个，如下所示。

```
module test (in,out);
input in;
output out;
assign out = !in;
endmodule
module top (in0,out0);
input in0;
output out0;
test test1(
.in(in0),
.out(out0)
);
endmodule
```

EX3771

WARN (EX3771): <modu> instantiation should have an instance name

若设计中使用实例化模块<modu>但没有给出名称时，综合工具会给出上述警告信息。如下案例 test 模块实例化时没有定义名称，综合时会给出一个默认名称。

```
module test (in,out);
input in;
```

```
output [1:0]out;
assign out = in+1'b1;
endmodule
module top (top_in,top_out);
input top_in;
output [1:0]top_out;
test (
.in(top_in),
.out(top_out)
);
endmodule
```

Action

若要消除上述警告，需要定义实例化模块名称，如下所示。

```
module test (in,out);
input in;
output [1:0]out;
assign out = in+1'b1;
endmodule
module top (top_in,top_out);
input top_in;
output [1:0]top_out;
test test1(
.in(top_in),
.out(top_out)
);
endmodule
```

EX3779

WARN (EX3779):<signal> should be on the sensitivity list

若设计中使用 always 语句但敏感信号列表缺少<signal>时，综合工具会给出上述警告信息。如下案例 in0 和 in1 是敏感信号，应该加到 always 敏感列表内，否则综合时会自动添加。

```
module top (sel,in0,in1,out);
input sel,in0,in1;
output reg out;
always@(sel /*or in0 or in1*/ )
    if(sel == 1'b0)
        begin
            out <= in0;
        end
```

```
    else
        begin
            out <= in1;
        end
endmodule
```

Action

若要消除上述警告，需要将对应信号添加到 always 敏感列表内，如下所示。

```
module top (sel,in0,in1,out);
input sel,in0,in1;
output reg out;
always@(sel or in0 or in1)
    if(sel == 1'b0)
        begin
            out <= in0;
        end
    else
        begin
            out <= in1;
        end
endmodule
```

EX3780

WARN (EX3780): Using initial value of <vari> since it is never assigned

若设计中定义一个寄存器变量<vari>，但只赋初始值作为常量使用，综合工具会给出此条警告信息。如下案例寄存器 tmp 仅作为 1’b0 使用。

```
module test(in,out);

input in;

output reg out;

reg tmp;

initial begin

        tmp = 0;

    end

always@(in or tmp)

    if(in==tmp)
```

```
        out <= in;
    else
        out <= !in;
endmodule
```

Action

若要消除上述警告，需要将无用的寄存器 tmp 替换成 1’b0，如下所示。

```
module test(in,out);
input in;
output reg out;
always@(in)
    if(in==1'b0)
        out <= in;
    else
        out <= !in;
endmodule
```

EX3784

WARN (EX3784): Index <width> is out of range <range> for <port>

若设计中使用<width>超过定义数据位宽范围<range>的<port>，综合工具会给出上述警告信息，如下案例 out 的定义位宽范围是 0 到 1，而却给 out[2] 赋值，综合时会忽略。

```
module test (in,out);
input [2:0]in;
output [1:0]out;
assign out[0] = in[0];
assign out[2] = in[1] & !in[2];
assign out[1] = in[1] & in[2];
endmodule
```

Action

若要消除上述警告，需要调整 out 宽度，或删除未定义宽度的使用，如下所示。

```
module test (in,out);
input [2:0]in;
```

```
output [2:0]out;
assign out[0] = in[0];
assign out[2] = in[1] & !in[2];
assign out[1] = in[1] & in[2];
endmodule
```

EX3786

ERROR (EX3786): Assignment to input <port>

若设计中存在给输入端口<port>赋值的情况，综合工具会给出上述错误信息。如下案例输入端口 d 被其他输入端口驱动。

```
module test(b,c,d,f);
input b,c,d;
output f;
assign d = c&b;
assign f = b&d;
endmodule
```

Action

若要消除上述错误，需要将此情况删除，如下所示。

```
module test(b,d,f);
input b,d;
output f;
assign f = b&d;
endmodule
```

EX3791

WARN (EX3791): Expression size <size> truncated to fit in target size <tarSize>

若设计中赋值操作前数据宽度<tarSize>和操作后<size>不同，综合工具会给出上述警告信息。如下案例 out 的数据宽度是 1，in0&in1 的数据宽度是 3，此时 in0&in1 的额外宽度无效。

```
module top (in0,in1,clk,out);
input [2:0]in0,in1;
input clk;
output reg out;
always @( posedge clk)
    begin
        out <= in0&in1;
    end
```

```
endmodule
```

Action

若要消除上述警告，可以调整 out 的宽度或 in0 和 in1 的宽度，如下所示。

```
module top (in0,in1,clk,out);
input [2:0]in0,in1;
input clk;
output reg [2:0] out;
always @( posedge clk)
    begin
        out <= in0&in1;
    end
endmodule
```

EX3792

WARN (EX3792): Literal value truncated to fit in <num> bits

若设计中定义了 1 个超出范围的 parameter 值<num>，综合工具会给出此条警告信息。如下案例 LUT2 的 INIT 值范围是 4’h0 到 4’hF，而 4’h14 本身就是非法值，综合时会取 2 进制后四位，为 4’h4。

```
module top (in0,in1,out);
input in0,in1;
output out;
LUT2 lut2(
.I0(in0),
.I1(in1),
.F(out)
);
defparam lut2.INIT = 4'h14;
endmodule
```

Action

若要消除上述警告，需要将 parameter 值修改到一个合理的范围内，如下所示。

```
module top (in0,in1,out);
input in0,in1;
output out;
LUT2 lut2(
.I0(in0),
.I1(in1),
.F(out)
);
```

```
defparam lut2.INIT = 4'h4;
endmodule
```

EX3794

ERROR (EX3794): Overwriting previous definition of module<modu>

若设计中定义了两个同名的<modu>，综合工具会给出此条错误信息，需要修改一个<modu>的名称。

```
module test (in0,in1,out);
input in0,in1;
output out;
assign out = in0 & !in1;
endmodule
module test (data,out);
input data;
output out;
assign out = !data;
endmodule
```

Action

若要消除上述错误，需要修改一个 module 的名称，如下所示。

```
module test (in0,in1,out);
input in0,in1;
output out;
assign out = in0 & !in1;
endmodule
module test0 (data,out);
input data;
output out;
assign out = !data;
endmodule
```

EX3818

ERROR (EX3818): <inst> expects <num> arguments

若综合工具报出上述错误信息，则说明实例化模块<inst>时给出的端口实例化数量<num>超过所需要的，如下案例中 ALU 需要 6 个端口，而实际上给出了 7 个。

```
module alu_1bit(a,b,din1,din2,sum,cout);
input din1,din2,a,b;
```

```
output cout,sum;
ALU sum_cry_0_0 (cout, sum, 0, din2, din1, a, b);
defparam sum_cry_0_0.ALU_MODE=0;
endmodule
```

Action

若要消除上述错误，需要调整参数个数，如下所示。

```
module alu_1bit(a,b,din1,din2,sum,cout);
input din1,din2,a,b;
output cout,sum;
ALU sum_cry_0_0 (cout, sum, din2, din1, a, b);
defparam sum_cry_0_0.ALU_MODE=0;
endmodule
```

EX3827

WARN (EX3827): Full_case directive is effective : might cause synthesis - simulation differences

若设计中使用 full_case 语法，综合工具会给出上述警告信息，可能会造成仿真不等价。

```
module top (sel,in0,out);

input sel,in0;

output reg out;

always@(sel or in0)

    begin

        case(sel)/*synthesis full_case*/

            1'b0:

                begin

                    out <= in0;

                end

        endcase

    end

endmodule
```

Action

此警告是添加 full_case 语句的提示信息，会减少 case 语句的无关条件

逻辑电路，若要消除上述警告，需要补全其他 case 条件。

EX3829

ERROR (EX3829): Port <port> is not defined

若综合工具报出上述错误信息，则说明声明的端口<port>没有添加到端口列表中。如下案例中 out1 端口有声明但是没有在模块定义的端口列表中。

```
module test (in,out);
input in;
output out;
output out1;
assign out = !in;
endmodule
```

Action

若要消除上述错误，可以把 out1 的声明删除，如下所示。

```
module test (in,out);
input in;
output out;
assign out = !in;
endmodule
```

EX3833

ERROR (EX3833): If-condition does not match any sensitivity list edge

若设计中使用 always 语句包含多个敏感信号，且内部的 if 条件的信号不在敏感信号列表内时，综合工具会给出此条错误信息。如下案例 clear 信号不在敏感信号列表内，这种情况无法创建出标准类型的触发器。

```
module top (in,out,clk1,clk2,clear);
input in,clk1,clk2,clear;
output reg out;
always @(posedge clk1 or posedge clk2)
begin
    if(clear)
        out <= 1'b0;
    else
        out <= in;
end
endmodule
```

Action

若要消除上述错误，需要移除 clk2，并将 clear 加入敏感列表，如下所示。

```
module top (in,out,clk1,clear);
input in,clk1,clear;
output reg out;
always @(posedge clk1 or posedge clear)
begin
    if(clear)
        out <= 1'b0;
    else
        out <= in;
end
endmodule
```

EX3834

WARN (EX3834): Case condition never applies

若设计中使用 case 语句但有些情况永远不会出现时，综合工具会给出此条警告信息。如下案例中 3'b101 位宽与 sel 信号不同，此情况永远不会出现。

```
module top (sel,in0,in1,out);
input sel,in0,in1;
output reg out;
always@(sel or in0 or in1 )
    begin
        case(sel)
            1'b0:
                begin
                    out <= in0;
                end
            1'b1:
                begin
                    out <= in1;
                end
3'b101:
                begin
                    out <= 1'b0;
                end
        endcase
    end
endmodule
```

Action

若要消除上述警告，需要将无用 case 条件移除，如下所示。

```
module top (sel,in0,in1,out);
input sel,in0,in1;
output reg out;
always@(sel or in0 or in1 )
    begin
        case(sel)
            1'b0:
                begin
                    out <= in0;
                end
            1'b1:
                begin
                    out <= in1;
                end
        endcase
    end
endmodule
```

EX3858

WARN (EX3858): System task <task> ignored for synthesis

综合不支持<task>语句，若设计中存在<task>语句。综合时会忽略，并给出此条警告信息。

EX3863

ERROR (EX3863): Syntax ERROR near <object>

若设计中<object>周围存在语法错误，综合工具会给出此条错误信息。如下案例第一行，最后要加分号。

```
module test (in,out)
input in;
output out;
assign out = !in;
endmodule
```

Action

若要消除上述错误，请检查语法错误原因，如下所示。

```
module test (in,out);
input in;
```

```
output out;
assign out = !in;
endmodule
```

EX3864

WARN (EX3864): <port> was previously declared with a different range

若设计中存在定义一个端口<port>，但又声明为不同宽度的连线或寄存器的情况，综合工具会给出上述警告信息。如下案例 out 被定义为不同的宽度的连线，综合时会以连线宽度为准。

```
module top(in,out);

input in;

output [1:0]out;

wire [2:0] out = 3'b0+in;

endmodule
```

Action

若要消除上述警告，需要将保持同名端口和连线宽度保持一致，如下所示。

```
module top(in,out);

input in;

output [2:0]out;

wire [2:0] out = 3'b0+in;

endmodule
```

EX3872

ERROR (EX3872): <port> is not declared

若综合工具报出上述错误信息，则说明端口<port>没有声明端口类型，如下案例中 out1 没有声明端口类型。

```
module test (in,out,out1);
input in;
output out;
assign out = !in;
endmodule
```

Action

若要消除上述错误需要将未声明端口类型的端口删除（或添加 port 类型声明），如下所示。

```
module test (in,out);
input in;
output out;
assign out = !in;
endmodule
```

EX3875

ERROR (EX3875) : No definition for port <port>

若综合工具报出上述错误信息，则说明端口<port>只有声明，没有定义端口方向。如下示例，在 out 声明中没有定义端口方向。

```
module test(in,out);
input in;
assign out = !in;
endmodule
module top (top_in,top_out);
input top_in;
output top_out;
test test1(
.in(top_in),
.out(top_out)
);
endmodule
```

Action

若要消除上述错误，需要定义 out 端口方向，如下所示。

```
module test(in,out);
input in;
output out;
assign out = !in;
endmodule
module top (top_in,top_out);
input top_in;
output top_out;
test test1(
.in(top_in),
.out(top_out)
);
```

```
endmodule
```

EX3900

ERROR (EX3900): Procedural assignment to a non-register <nreg> is not permitted

若综合工具报出上述错误信息，则说明有给非寄存器类型<nreg>进行非阻塞赋值的情况。如下案例中 out 声明应为寄存器类型。

```
module top (in,out,clk);
input in,clk;
output out;
wire out;
always @( posedge clk)
    begin
        out <= in;
    end
endmodule
```

Action

若要消除上述错误,需要将 out 声明改为寄存器类型，如下所示。

```
module top (in,out,clk);
input in,clk;
output out;
reg out;
always @( posedge clk)
    begin
        out <= in;
    end
endmodule
```

EX3902

ERROR (EX3902): Port <port> is already defined

若设计中端口<port>被重复定义，综合工具报出上述错误信息。示例如下，案例中输出端口 out 被重复定义。

```
module top (in,out);
input in;
output out;
output out;
assign out = !in;
endmodule
```

Action

若要消除上述错误，需要将重复定义移除，如下所示。

```
module top (in,out);
input in;
output out;
assign out = !in;
endmodule
```

EX3907

ERROR (EX3907): Parameter <para>is not defined in this module

若设计中存在一个 Instance 设置了当前模块未定义的<para>时，综合工具给出上述错误信息。可根据错误信息提示的参数名和文件行信息进行查找及修改。示例如下，ins1 设置了 INIT_0 的参数，但此参数不是 module DFF 的参数。

```
module test(a,clk,out);
input a,clk;
output out;
DFF ins1(
    .D(a),
    .CLK(clk),
    .Q(out)
);
defparam ins1.INIT=1'b0;
defparam ins1.INIT_0=8'hEF;
endmodule
```

Action

若要消除上述错误，需要将此参数删除，如下所示。

```
module test(a,clk,out);
input a,clk;
output out;
DFF ins1(
    .D(a),
    .CLK(clk),
    .Q(out)
);
defparam ins1.INIT=1'b0;
endmodule
```

EX3916

WARN (EX3916): No support for synthesis of mixed edge and level triggers. Assume level triggers only.

若设计中 always 语句中同时包含边沿触发信号和电平触发信号时。综合工具报出上述警告信息，此时边沿触发信号会被忽略。

```
module top (in,out,clk,clear);
input in,clk,clear;
output reg out;
always @(posedge clk or clear)
    if(clear)
        begin
            out <= 1'b0;
        end
    else
        begin
            out <= in;
        end
endmodule
```

Action

若要消除上述警告，需要将 clear 信号从敏感列表移除，如下所示。

```
module top (in,out,clk,clear);
input in,clk,clear;
output reg out;
always @(posedge clk)
    if(clear)
        begin
            out <= 1'b0;
        end
    else
        begin
            out <= in;
        end
endmodule
```

EX3927

ERROR (EX3927): Module<modu> remains a black box, due to ERRORs in its contents

此信息与其他错误信息同时出现。若综合工具报出上述错误信息，则说

明<modu>中存在其他错误，此时综合报错退出。

EX3928

ERROR (EX3928): Module <modu> ignored due to previous ERRORs

此信息与其他错误信息同时出现，若综合工具报出上述错误信息，则说明<modu>中存在其他错误，此时综合报错退出。

EX3937

ERROR (EX3937): Instantiating unknown module <modu>

若设计中实例化一个非高云原语的模块<modu>，但没有模块定义，综合工具会给出此条错误信息。示例如下，案例 test 没有模块定义。

```
module top (in,out);
input in;
output out;
test test1(
.in0(in),
.out0(out)
);
endmodule
```

Action

若要消除上述错误，需要添加<modu>的模块定义，模块内部实现可以为空，若内部实现为空，则会被转化为黑盒子。如下所示。

```
module top (in,out);
input in;
output out;
test test1(
.in0(in),
.out0(out)
);
endmodule
module test(in0,out0);
input in0;
output out0;
assign out0 = !in0;
endmodule
```

EX3945

ERROR (EX3945): Incorrect use of predefined macro <include>. Expected <filePath>

若设计中错误使用<include>语句指定文件路径<filePath>时，综合工具会给出上述错误信息。如下案例 include 语句的文件路径两端没有加双引号。

```
`include param.v;

module top(in,sel,out);

input in,sel;

output reg [size:0]out;

assign out = in+sel;

endmodule

//param.v 文件内容

/*

parameter size = 2;

*/
```

Action

若要消除上述错误，需要在 include 语句的文件路径两端加双引号，如下所示。

```
`include "param.v";

module top(in,sel,out);

input in,sel;

output reg [size:0]out;

assign out = in+sel;

endmodule

//param.v 文件内容

/*

parameter size = 2;

*/
```

EX3983

WARN (EX3983): Case condition never applies due to comparison with x or z

若设计中使用 case 语法时包含 X 和 Z 值时，综合工具会给出此条警告信息。如下案例 case 语句中包含 X 和 Z 的两种情况，综合时会忽略这两种情况。

```
module top (sel,in0,in1,out);
input sel,in0,in1;
output reg out;
always@(sel or in0 or in1 )
    begin
        case(sel)
            1'b0:
                begin
                    out <= in0;
                end
            1'b1:
                begin
                    out <= in1;
                end
1'bX:
                begin
                    out <= 1'b0;
                end
1'bZ:
                begin
                    out <= 1'b1;
                end
        endcase
    end
endmodule
```

Action

若要消除上述警告，需要将包含 X 和 Z 值的情况删除，如下所示。

```
module top (sel,in0,in1,out);
input sel,in0,in1;
output reg out;
always@(sel or in0 or in1 )
    begin
        case(sel)
            1'b0:
```

```
                begin
                    out <= in0;
                end
            1'b1:
                begin
                    out <= in1;
                end
        endcase
    end
endmodule
```

EX3988

WARN(EX3988): Cannot open file <file>

若设计中读取相应的配置文件<file>，但配置文件<file>不存在或没有权限打开时，综合工具会给出上述警告信息。如下案例$readmemh 语法对应的 initvalue 文件不存在或没有权限打开时，会报出上述警告。

```
module test(in,clk,addr_in,addr_out,out);

input in,clk;

input addr_in;

input addr_out;

output out;

reg [7:0]mem [7:0];

 always @ (posedge clk)

    mem[addr_in] <= in;

 assign out = mem[addr_out];

initial begin

$readmemh("initvalue", mem);

  end

endmodule
```

Action

若要消除上述警告，需要 initvalue 文件存在并有读权限。

IF0003

ERROR (IF0003): Cannot infer <signal> due to multiple write

clocks

RAM Inference，最多只支持两个 clock 写入，如果优化之后还超出两个写入端，综合工具会报出上述错误。

```
    module normal5(data_out0,data_out1, data_in0,
data_in1,data_in2,addr,addr0, addr1,addr2,clk0,clk1,clk2,ce, wre,rst);
    input [2:0]data_in0;
    input [2:0]data_in1;
    input [2:0]data_in2;
    input [3:0]addr,addr0, addr1,addr2;
    input clk0,clk1,clk2,wre,ce,rst;
    reg [2:0] mem [7:0] ;
    output reg [2:0] data_out0;
    output reg [2:0] data_out1;
    always@(posedge clk0)
         if(ce==1 & wre == 0)
            data_out0 <= mem[addr0];
    always@(posedge clk1)
         if(ce==1 & wre == 0)
            data_out1 <= mem[addr1];
     always @(posedge clk0)
        if (ce & wre) mem[addr0] <= data_in0;
     always @(posedge clk1)
        if (ce & wre) mem[addr1] <= data_in1;
     always @(posedge clk1)
        if (ce & wre) mem[addr2] <= data_in2;
    endmodule
```

Action

不可以同时往同一块内存写超过两组数据，如果优化之后还超出两个写入端，则没有对应高云原语支持此功能，请修改 rtl 设计。

Place & Route 用户消息

CT1000

WARN (CT1000):<file>:<line> | This constraint of <name>is defined again, so this will overwrite the previous

存在重复的约束，仅保留后者的约束内容。

```
    INS_LOC uut R3C4;

    INS_LOC uut R4C5;
```

Action

修改约束文件，删除重复的约束。

```
INS_LOC uut R4C5;
```

CT1003

WARN (CT1003) :<file>:<line> | Group(<name>) location is already defined, so this will overwrite the previous

存在组约束的重复定义，仅保留后者的约束内容。

```
GROUP grp = {“ins1” “ins2”}

GRP_LOC grp R3C[3:5];

GRP_LOC grp R[4:5]C8;
```

Action

修改约束文件，删除重复的约束。

```
GROUP grp = {“ins1” “ins2”}

GRP_LOC grp R[4:5]C8;
```

CT1005

WARN (CT1005):Conflicting multiple constraints specified for location of Instance <name>(type: <type>); Or constrained location for the Instance is not available; Or constrained location type is not matched with the instance

多个约束存在冲突，或约束位置不合理，或约束位置与原语不匹配。

Action

修改约束文件，将约束对象约束到合理的位置，并避免和其它约束的冲突。

CT1007

WARN (CT1007):There is no intersection between multiple group constraints specified for instance <name>

约束对象存在于多个约束组中，但约束组之间不存在共同的约束位置，导致该约束对象无正确的约束位置。

```
GROUP  grp1={“ins1” “ins2” “ins3”};

GRP_LOC  grp1  R2C[5:6];
```

```
GROUP  grp2 = {“ins1” “ins4”};

GRP_LOC  grp2  R4C[5:6];
```

Action

修改约束文件，避免将一个约束对象同时放到多个约束组中。

```
GROUP  grp1={“ins2”“ins3”};

GRP_LOC  grp1  R2C[5:6];

GROUP  grp2 = {“ins1” “ins4};

GRP_LOC  grp2  R4C[5:6];
```

CT1097

WARN (CT1097) :<file>:<line> | Please define group <name> first before define the constraint at line <number>

对未定义的约束组进行位置约束。

```
GRP_LOC  grp1  R2C[5:6];
```

Action

在对约束组进行位置约束之前，需定义约束组。

```
GROUP  grp1={“ins2”“ins3”};

GRP_LOC  grp1  R2C[5:6];
```

CT1098

WARN (CT1098) :<file>:<line> | Group name <name> is already defined

存在约束组的重复定义。

```
GROUP  grp1={“ins2”“ins3”};

REL_GROUP  grp1={“ins4”“ins5”};

GRP_LOC  grp1  R2C[5:6];
```

Action

修改约束文件，避免重复约束组的定义。

```
GROUP  grp1={“ins4”“ins5”};

GRP_LOC  grp1  R2C[5:6];
```

CT1101

WARN (CT1101) :<file>:<line> | Location column <number> is out of chip range(<maxColumn>)

约束位置信息中的列超出了芯片的范围。

Action

修改约束位置信息，使列不超出芯片的范围。

CT1102

WARN (CT1102): <file>:<line> | Location row <number> is out of the chip range(<maxRow>)

约束位置信息中的行超出了芯片的范围。

Action

修改约束位置信息，使行不超出芯片范围。

CT1108

WARN (CT1108) :<file>:<line> | Illegal port attribute value specified <attribute> = <value> on <instName>

不正确的属性约束，属性值与属性不匹配。

```
IO_PORT  bufIns  DRIVE=20;
```

Action

修改该属性的属性值。

```
IO_PORT  bufIns  DRIVE=8;
```

CT1111

WARN (CT1111): Instance <name>(<type>) constrained to unsuitable location

将约束对象约束到了不合理的约束位置。

```
INS_LOC dll_inst_2 PLL_R;
```

Action

依据约束对象的类型，将其约束到对应的约束位置处。

```
INS_LOC  dll_inst_2  DLL_BR;
```

CT1112

WARN (CT1112) :<file>:<line> | Invalid range location <location>, please constrainedin the same side

进行区域位置约束时，区域的起始位置和结束位置应在相同的边上。

```
INS_LOC bufIns IOR4:IOL9;
```

Action

修改区域约束的起始位置或结束位置，使其在相同的边上。

```
INS_LOC bufIns IOR4:IOR9;
```

CT1113

WARN (CT1113) :<file>:<line> | Cannot find pad location <pin> in current package

当前封装不存在该约束位置。

Action

修改约束位置，确保位置信息对当前封装是可用的。

CT1115

WARN (CT1115): Attribute <name> can only be set when the port is located to bank <index>. Please set the corresponding location constraint of port <portName>

对接口进行属性约束时，应先进行位置约束。

```
IO_PORT i0 IO_TYPE=RSDS25E DIFF_RESISTOR=ON;
```

Action

先对接口进行位置约束，然后进行属性约束。

```
IO_LOC i0 IOT4;

IO_PORT i0 IO_TYPE=RSDS25E DIFF_RESISTOR=ON;
```

CT1116

WARN (CT1116): Attribute <name> can only be set when the port is located to bank <index>. Please set the corresponding location constraint of port <portName> or <portName>

对差分接口进行属性约束时，应先进行位置约束。

```
IO_PORT I IO_TYPE=RSDS25E DIFF_RESISTOR=ON;
```

Action

先对接口 IB 或其差分接口 I 进行位置约束，然后进行属性约束。

```
IO_LOC IB IOT4;

IO_PORT I IO_TYPE=RSDS25E DIFF_RESISTOR=ON;
```

或：

```
IO_LOC I IOT4;

IO_PORT I IO_TYPE=RSDS25E DIFF_RESISTOR=ON;
```

CT1117

WARN (CT1117): Attribute <name> can only be set when the port is located to bank <index>, but the constraint location of port <portName> include other bank

属性的约束值与约束位置不匹配。

```
IO_LOC i0 IOB4;

IO_PORT i0 IO_TYPE=RSDS25E DIFF_RESISTOR=ON;
```

Action

修改属性约束或约束位置。

```
IO_LOC i0 IOT4;

IO_PORT i0 IO_TYPE=RSDS25E DIFF_RESISTOR=ON;
```

CT1118

WARN (CT1118): Attribute <name> can only be set when the port is located to bank <index>, but the constraint location of port <portName> or <portName> include other bank

属性的约束值与约束位置不匹配。

```
IO_LOC I IOB4;

IO_PORT I IO_TYPE=RSDS25E DIFF_RESISTOR=ON;
```

Action

修改对接口 I 或其差分接口 IB 的属性约束或位置约束。

```
IO_LOC I IOT4;

IO_PORT I IO_TYPE=RSDS25E DIFF_RESISTOR=ON;
```

或：

```
IO_LOC IB IOT4;

IO_PORT I IO_TYPE=RSDS25E DIFF_RESISTOR=ON;
```

FS1008

WARN (FS1008):Device <device type> is not supported AES encryption, please uncheck in bitstream configurations

当前 Device 不支持 AES 加密。

Action

修改配置选项，取消加密选项。

FS2001

ERROR (FS2001) :Cannot read corrupted fse file

读取 fse 失败。

Action

使用与当前软件匹配的 fse 文件，不要删除或修改 fse 文件。

PA1000

WARN (PA1000):Dangling net <netName> in module <moduleName> has no source instance

模块中的线没有源。

```
module test (i0,i1,i2,i3,out);

input i0;

input i1;

input i2;

input i3;

output out;

wire i4;

LUT4 uut (
```

```
        .I0(i0),
        .I1(i1),
        .I2(i2),
        .I3(i4),
        .F(out)
    );
    endmodule
```

Action

建立正确的连接关系，确保每一个连线存在信号源。若设计中该连线应悬空，请忽略该警告信息。

```
    module test (i0,i1,i2,i3,out);
    input i0;
    input i1;
    input i2;
    input i3;
    output out;
    LUT4 uut (
        .I0(i0),
        .I1(i1),
        .I2(i2),
        .I3(i3),
        .F(out)
    );
    endmodule
```

PA1001

WARN (PA1001): Dangling net <netName>(source:<instanceName>) in module <moduleName> has no destination

指定模块中的连线没有连接目的原语。

```
    module test (i0,i1,i2,i3,out);
```

```
input i0;
input i1;
input i2;
input i3;
output out;
wire out_c;
LUT4 uut (
    .I0(i0),
    .I1(i1),
    .I2(i2),
    .I3(i3),
    .F(out_c)
);
endmodule
```

Action

建立正确的连接关系，确保连线能够有信号的终点。若设计中该连线应悬空，请忽略该警告信息。

```
module test (i0,i1,i2,i3,out);
input i0;
input i1;
input i2;
input i3;
output out;
wire out_c;
LUT4 uut (
    .I0(i0),
    .I1(i1),
    .I2(i2),
    .I3(i3),
    .F(out_c)
```

```
);
OBUF buf_ins (
    .I(out_c),
    .O(out)
);
endmodule
```

PA1002

WARN (PA1002): <file>:<line> | Invalid parameterized value <value>(<parameter>) specified for instance <instanceName>

指定位置处的原语设置了错误的参数值。

Action

请为原语的参数配置正确的参数值。

PA1008

WARN (PA1008): <file>:<line> | Object <name> is already defined

在指定位置处，存在连线或接口的重复定义。

```
module test (i0,i1,i2,i3,out);
input i0;
input i1;
input i2;
input i3;
input i3;
output out;
LUT4 uut (
    .I0(i0),
    .I1(i1),
    .I2(i2),
    .I3(i3),
    .F(out)
);
```

```
endmodule
```

Action

删除设计文件中连线或接口的重复定义。

```
module test (i0,i1,i2,i3,out);
input i0;
input i1;
input i2;
input i3;
output out;
LUT4 uut (
    .I0(i0),
    .I1(i1),
    .I2(i2),
    .I3(i3),
    .F(out)
);
endmodule
```

PA1010

WARN (PA1010): <file>:<line> | Dangling pin(<name>) is not connect with net

在指定位置处的原语引脚没有建立连接关系。

```
module test (i0,i1,i2,i3,out);
input i0;
input i1;
input i2;
input i3;
output out;
LUT4 uut (
    .I0(i0),
```

```
        .I1(i1),
        .I2(i2),
        .I3(i3),
        .F()
    );
    endmodule
```

Action

为器件的引脚建立正确的连接关系。若设计文件中该引脚应悬空，请忽略该警告信息。

```
    module test (i0,i1,i2,i3,out);
    input i0;
    input i1;
    input i2;
    input i3;
    output out;
    LUT4 uut (
        .I0(i0),
        .I1(i1),
        .I2(i2),
        .I3(i3),
        .F(out)
    );
    endmodule
```

PA2000

ERROR (PA2000): <file>:<line> | Syntax error near token <name>

指定位置处存在语法错误。

```
    module test (i0,i1,i2,i3,out);
    input i0;
    input i1;
```

```
input i2;
input i3;
ouput out;
LUT4 uut (
    .I0(i0),
    .I1(i1),
    .I2(i2),
    .I3(i3),
    .F(out)
);
endmodule
```

Action

依据提示信息，查找设计文件中存在的语法错误并纠正。

```
module test (i0,i1,i2,i3,out);
input i0;
input i1;
input i2;
input i3;
output out;
LUT4 uut (
    .I0(i0),
    .I1(i1),
    .I2(i2),
    .I3(i3),
    .F(out)
);
endmodule
```

PA2001

ERROR (PA2001): <file>:<line> | Module <moduleName> is

already defined

存在模块的重复定义。

```
module test (i0,i1,i2,i3,out);
input i0;
input i1;
input i2;
input i3;
output out;
LUT4 uut (
    .I0(i0),
    .I1(i1),
    .I2(i2),
    .I3(i3),
    .F(out)
);
endmodule

module test (I0,I1,OUT);
input I0;
input I1;
output OUT;
LUT2 uut (
    .I0(I0),
    .I1(I1),
    .F(OUT)
);
endmodule
```

Action

修改设计文件中重复的模块名。

```
module test (i0,i1,i2,i3,out);
```

```
input i0;
input i1;
input i2;
input i3;
output out;
LUT4 uut (
    .I0(i0),
    .I1(i1),
    .I2(i2),
    .I3(i3),
    .F(out)
);
endmodule

module testLut2 (I0,I1,OUT);
input I0;
input I1;
output OUT;
LUT2 uut (
    .I0(I0),
    .I1(I1),
    .F(OUT)
);
endmodule
```

PA2004

ERROR (PA2004) :<file> | In module <name>: Net <netName> driven by multiple source instances

指定模块中的连线存在多个信号源。

```
module test (i0,i1,i2,i3,out, out1);
```

```
input i0;
input i1;
input i2;
input i3;
output out;
output out1;
wire out_c;
LUT4 uut (
    .I0(i0),
    .I1(i1),
    .I2(i2),
    .I3(i3),
    .F(out_c)
);
LUT2 uut2 (
    .I0(i0),
    .I1(i1),
    .F(out_c)
);
OBUF bufIns (
    .I(out_c),
    .O(out)
);
endmodule
```

Action

依据提示信息，修改指定模块中的连接关系。

```
module test (i0,i1,i2,i3,out,out1);
input i0;
input i1;
input i2;
```

```
input i3;
output out;
output out1;
wire out1_c;
wire out_c;
LUT4 uut (
    .I0(i0),
    .I1(i1),
    .I2(i2),
    .I3(i3),
    .F(out_c)
);
LUT2 uut2 (
    .I0(i0),
    .I1(i1),
    .F(out1_c)
);
OBUF bufIns (
    .I(out_c),
    .O(out)
);
OBUF bufIns1 (
    .I(out1_c),
    .O(out1)
);
endmodule
```

PA2009

ERROR (PA2009): The port <name> connected to <instName>(instType) defined error direction which should be <portType> according to connection

端口的类型与接口的连接存在冲突。

```
module test (i0,i1,i2,i3,out);
input i0;
input i1;
input i2;
input i3;
input out;
LUT4 uut (
    .I0(i0),
    .I1(i1),
    .I2(i2),
    .I3(i3),
    .F(out)
);
endmodule
```

Action

修改接口的连接关系或接口的类型。

```
module test (i0,i1,i2,i3,out);
input i0;
input i1;
input i2;
input i3;
output out;
LUT4 uut (
    .I0(i0),
    .I1(i1),
    .I2(i2),
    .I3(i3),
    .F(out)
);
```

```
endmodule
```

PA2014

ERROR (PA2014): Pin(<name>) of <instName>(<instType>) does not connect to port

指定实例化模块的引脚没有连接模块的端口。

```
module test (i0,i1,i2,i3,out);
input i0;
input i1;
input i2;
input i3;
output out;
wire i0_c;
wire VCC;
wire io;
LUT4 uut (
    .I0(i0_c),
    .I1(i1),
    .I2(i2),
    .I3(i3),
    .F(out)
);
IOBUF bufIns (
    .I(i0),
    .O(i0_c),
.IO(io),
    .OEN(VCC)
);
VCC vcc (
    .V(VCC)
```

```
);
endmodule
```

Action

修改连接关系，建立实例化模块引脚与模块端口的连接。

```
module test (i0,i1,i2,i3,out,io);
input i0;
input i1;
input i2;
input i3;
output out;
inout io;
wire i0_c;
wire VCC;
wire io;
LUT4 uut (
    .I0(i0_c),
    .I1(i1),
    .I2(i2),
    .I3(i3),
    .F(out)
);
IOBUF bufIns (
    .I(i0),
    .O(i0_c),
    .IO(io),
    .OEN(VCC)
);
VCC vcc (
    .V(VCC)
);
```

```
endmodule
```

PA2017

ERROR (PA2017): The number(<value>) of <instType> in the design exceeds the resource limit(<maxValue>)of current device

设计文件中<instType>类型原语的数量超出了该器件的总数。

Action

修改设计文件，减少该类型原语的数量，或采用资源量更大的器件。

PA2024

ERROR (PA2024): The number(<value>) of ports exceeds the resource limit <maxValue> regularI/Os(include <value> dedicated I/Os) and <value> shared I/Os of current device

顶层模块的接口数量超出器件接口总数。

Action

修改设计文件，或采用其它封装类型，或采用资源量更大的器件。

PA2025

ERROR (PA2025): No <instType> resource in current device

设计文件中含有本器件不支持的资源。

Action

修改设计文件，或采用支持该资源的其它系列的器件。

PA2039

ERROR (PA2039): Net <name>is used in module <moduleName> but not declared in wire list

使用未声明的连线。

```
module test (i0,i1,i2,i3,out);

input i0;

input i1;

input i2;

input i3;
```

```
output out;
LUT4 uut (
    .I0(i0),
    .I1(i1),
    .I2(i2),
    .I3(i3),
    .F(out_c)
);
OBUF obufIns (
    .O(out),
.I(out_c)
);
endmodule
```

Action

声明该连线。

```
module test (i0,i1,i2,i3,out);
input i0;
input i1;
input i2;
input i3;
output out;
wire out_c;
LUT4 uut (
    .I0(i0),
    .I1(i1),
    .I2(i2),
    .I3(i3),
    .F(out_c)
);
OBUF obufIns (
```

```
    .O(out),
    .I(out_c)
);
endmodule
```

PA2054

ERROR (PA2054): <file>:<line> | <name> is already declared

设计文件中存在模块实例化名称重复。

```
module test (i0,i1,i2,i3,out);
input i0;
input i1;
input i2;
input i3;
output out;
wire out_c;
LUT4 uut (
    .I0(i0),
    .I1(i1),
    .I2(i2),
    .I3(i3),
    .F(out_c)
);
OBUF uut (
    .O(out),
    .I(out_c)
);
endmodule
```

Action

修改模块实例化名称，避免名称的重复。

```
module test (i0,i1,i2,i3,out);
```

```
input i0;
input i1;
input i2;
input i3;
output out;
wire out_c;
LUT4 uut (
    .I0(i0),
    .I1(i1),
    .I2(i2),
    .I3(i3),
    .F(out_c)
);
OBUF obufIns (
    .O(out),
    .I(out_c)
);
endmodule
```

PA2056

ERROR (PA2056): <file>:<line> | Error pin name(<name>) found in instance <instName>

实例化原语的引脚名称与该原语不匹配。

```
module test (i0,i1,i2,i3,out);
input i0;
input i1;
input i2;
input i3;
output out;
wire out_c;
```

```
LUT4 uut (
    .I0(i0),
    .I1(i1),
    .I2(i2),
    .I4(i3),
    .F(out_c)
);
OBUF bufIns (
    .O(out),
    .I(out_c)
);
endmodule
```

Action

检查指定原语的引脚，并修正错误的引脚名称。

```
module test (i0,i1,i2,i3,out);
input i0;
input i1;
input i2;
input i3;
output out;
wire out_c;
LUT4 uut (
    .I0(i0),
    .I1(i1),
    .I2(i2),
    .I3(i3),
    .F(out_c)
);
OBUF bufIns (
    .O(out),
```

```
    .I(out_c)
);
endmodule
```

PA2058

ERROR (PA2058): <file>:<line> | Error pin number within instance <name>(<type>) ofmodule<name>

实例化原语的引脚数量不正确。

```
module test (i0,i1,i2,i3,out);
input i0;
input i1;
input i2;
input i3;
output out;
wire out_c;
LUT4 uut (
    .I0({i0,i1}),
    .I1(i1),
    .I2(i2),
    .I3(i3),
    .F(out_c)
);
OBUF bufIns (
    .O(out),
    .I(out_c)
);
endmodule
```

Action

检查指定器件的引脚数量，删除多余的引脚，或增加缺少的引脚。

```
module test (i0,i1,i2,i3,out);
```

```
input i0;
input i1;
input i2;
input i3;
output out;
wire out_c;
LUT4 uut (
    .I0(i0),
    .I1(i1),
    .I2(i2),
    .I3(i3),
    .F(out_c)
);
OBUF bufIns (
    .O(out),
    .I(out_c)
);
endmodule
```

PA2066

ERROR(PA2066): <file>:<line> | Invalid parameter name <name> setting to object <instName>

原语设置不支持的属性。

```
module test (i0,i1,i2,i3,out);
input i0;
input i1;
input i2;
input i3;
output out;
wire out_c;
```

```
LUT4 uut (
    .I0(i0),
    .I1(i1),
    .I2(i2),
    .I3(i3),
    .F(out_c)
);
defparam uut.INIT_1=16'h0000;
OBUF bufIns (
    .O(out),
    .I(out_c)
);
endmodule
```

Action

修改指定原语的属性配置。

```
module test (i0,i1,i2,i3,out);
input i0;
input i1;
input i2;
input i3;
output out;
wire out_c;
LUT4 uut (
    .I0(i0),
    .I1(i1),
    .I2(i2),
    .I3(i3),
    .F(out_c)
);
defparam uut.INIT=16'h0000;
```

```
OBUF bufIns (
    .O(out),
    .I(out_c)
);
endmodule
```

PR0026

ERROR (PR0026): CLKOUTN pin of <name> is not connected to any other iologic

DHCENC 的 CLKOUTN pin 没有驱动任何 IOLOGIC。

Action

更改设计使 DHCENC 的 CLKOUTN pin 合理的驱动 IOLOGIC。

PR0027

ERROR (PR0027): Instance <name> connected to CLKIN pin of instance <name> is unsupported

该 Instance 不支持连接 PLL 或 DLL 的 CLKIN pin。

Action

更改设计该 Instance 不连接 PLL 或 DLL 的 CLKIN pin。

PR0028

ERROR (PR0028): Instance <name> connected to CLKFB pin of instance <name> is unsupported

该 Instance 不支持连接 PLL 的 CLKFB pin。

Action

更改设计该 Instance 不连接 PLL 的 CLKFB pin。

PR0029

ERROR (PR0029): Instance <name>(INS_DHCENC) cannot drive two CLKDIVs

DHCENC 不能同时驱动两个 CLKDIV。

Action

更改设计使 DHCENC 不同时驱动两个 CLKDIV。

PR1011

ERROR (PR1011): Failed to capture gao signal:<name>, because there'sno wiretoroutefor thesignal

GAO 抓取信号失败。如下测试案例中，ALU 链中的 signal ”c0_c”不可以绕线，所以当抓取 signal “c0_c”时，抓取失败。

```
    module test (i0, i1, i2, i3, o0, o1, o2);

    input i0, i1, i2, i3;

    output o0, o1, o2;

    wire i0_c, i1_c, i2_c, i3_c, c0_c, c1_c, s0_c, s1_c, GND;

    GND GND_C(.G(GND));

    IBUF ibuf_i0(.I(i0), .O(i0_c));

    IBUF ibuf_i1(.I(i1), .O(i1_c));

    IBUF ibuf_i2(.I(i2), .O(i2_c));

    IBUF ibuf_i3(.I(i3), .O(i3_c));

    ALU
alu_0(.I0(i0_c),.I1(i1_c),.I3(GND),.CIN(GND),.COUT(c0_c),.SUM(s0_c));

    defparam alu_0.ALU_MODE = 0;

    ALU
alu_1(.I0(i2_c),.I1(i3_c),.I3(GND),.CIN(c0_c),.COUT(c1_c),.SUM(s1_c));

    defparam alu_1.ALU_MODE = 0;

    OBUF obuf_sum0(.I(s0_c), .O(o0));

    OBUF obuf_sum1(.I(s1_c), .O(o1));

    OBUF obuf_cout(.I(c1_c), .O(o2));

    endmodule
```

Action

确认所抓取的信号是否可以抓取，当不可抓取时，请向前一级或向后一级逻辑抓取信号进行分析。如下的测试案例，可通过抓取 alu_0/alu_1 的 I0 和 I1 对应的信号，分析 signal “c0_c”。

```
    module test (i0, i1, i2, i3, o0, o1, o2);
```

```
    input i0, i1, i2, i3;

    output o0, o1, o2;

    wire i0_c, i1_c, i2_c, i3_c, c0_c, c1_c, s0_c, s1_c, GND;

    GND GND_C(.G(GND));

    IBUF ibuf_i0(.I(i0), .O(i0_c));

    IBUF ibuf_i1(.I(i1), .O(i1_c));

    IBUF ibuf_i2(.I(i2), .O(i2_c));

    IBUF ibuf_i3(.I(i3), .O(i3_c));

    ALU
alu_0(.I0(i0_c),.I1(i1_c),.I3(GND),.CIN(GND),.COUT(c0_c),.SUM(s0_c));

    defparam alu_0.ALU_MODE = 0;

    ALU
alu_1(.I0(i2_c),.I1(i3_c),.I3(GND),.CIN(c0_c),.COUT(c1_c),.SUM(s1_c));

    defparam alu_1.ALU_MODE = 0;

    OBUF obuf_sum0(.I(s0_c), .O(o0));

    OBUF obuf_sum1(.I(s1_c), .O(o1));

    OBUF obuf_cout(.I(c1_c), .O(o2));

    endmodule
```

PR1014

WARN(PR1014): Generic routing resource will be used to clock signal<name> by the specified constraint. And then it may lead to the excessive delay or skew

Gowin Router 检查到时钟信号的绕线信息中存在逻辑绕线，可能会导致时钟延迟或偏斜问题。如下的测试案例中（器件为 GW1N-4），将时钟信号”clk_c”的源约束到了非时钟端口，导致该时钟信号通过部分逻辑绕线资源。

```
    top.vm

    Module test_clk()

    input i0, i1

    output o0

    IBUF ibuf_data(.I(i0), .O(d_c));
```

```
IBUF ibuf_clk(.I(i1), .O(clk_c));

DFF dff_c(.D(d_c), .CLK(clk_c), .Q(q_c));

OBUF obuf_c(.I(q_c), .O(o0));

endmodule


top.cst

IO_LOC “ibuf_data” IOB18A;
```

Action

确认时钟信号的源是否为时钟信号源或者时钟信号所连接的端口的物理约束位置是否为时钟端口。如下的测试案例中（器件为GW1N-4），在GW1N-4中 IOB20A 为时钟端口，将时钟信号“clk_c”约束到该位置即可。

```
top.vm

Module test_clk()

input i0, i1

output o0

IBUF ibuf_data(.I(i0), .O(d_c));

IBUF ibuf_clk(.I(i1), .O(clk_c));

DFF dff_c(.D(d_c), .CLK(clk_c), .Q(q_c));

OBUF obuf_c(.I(q_c), .O(o0));

endmodule


top.cst

IO_LOC “ibuf_data” IOB20A;
```

PR2044

WARN (PR2044): FCLK port of <name> conflicts with FCLK port of <name> and <FCLK or HCLKIN> port of <name>

第一个 Instance 的 FCLK，与第二个 Instance 的 FCLK 和第三个 Instance 的 FCLK 或 HCLKIN 不共线。

Action

更改设计使之共线或更换其他可用位置。

PR2045

WARN (PR2045): FCLK port of <name>conflicts with FCLK port of <name>

第一个 Instance 的 FCLK 与第二个 Instance 的 FCLK 不共线。

Action

更改设计使之共线或更换其他可用位置。

PR2061

ERROR (PR2061): There is no position to place <name>

该 Instance 没有位置可放。

Action

改变能够影响该 Instance 布局的其他 Instance 的布局位置，使其有可用资源进行布局。

PR2062

ERROR (PR2062): Objects drived by CLKOUT pin of <name> must be placed on same side

被同一个 DHCENC 的 CLKOUT 驱动的所有 IOLOGIC 必须放在同一边。

Action

更换其他可用位置。

PR2063

ERROR (PR2063): Objects drived by CLKOUTN pin of <name> must be placed on same side with buffer <name>

被同一个 DHCENC 的 CLKOUTN 驱动的所有 IOLOGIC 必须与该 BUFFER 放在同一边。

Action

更换其他可用位置。

PR2064

ERROR (PR2064): Buffer<name> driving DHCENC must be placed to GCLK

驱动 DHCENC 的 BUFFER 必须放在 GCLK 的位置上。

Action

更换其他可用的 GCLK 位置。

PR2065

ERROR (PR2065): Buffer<name> driving DLLDLY must be placed to GCLK

驱动 DLLDLY 的 BUFFER 必须放在 GCLK 的位置。

Action

更换其他可用的 GCLK 位置。

PR2066

ERROR (PR2066): Iologics need more than two hclk on <chip side>

IOLOGICs 需要两个以上的 hclk，该边上的 hclk 资源不够。

Action

更换其他满足 hclk 资源且可用的位置。

PR2067

ERROR (PR2067): Instance <name> must have constraint

GW1N-9C 和 GW1NR-9C 要求该 Instance 必须有约束。

Action

对该 Instance 添加合理的约束进行布局。

PR2068

ERROR (PR2068): Instance <name> must have unique constraint

GW1N-9C 和 GW1NR-9C 要求该 Instance 必须有唯一约束。

Action

对该 Instance 添加唯一的约束进行布局。

PR2069

ERROR (PR2069): The constrainted location of <name> cannot be

IO BLOCK

该 Instance 的约束位置不能是 IOB。

Action

更换约束至其他非 IOB 的可用位置。

PR2070

ERROR (PR2070): Instance <name> connected to IODELAYC cannot be placed to bottom side

连接 IODELAYC 的 BUFFER，不能约束布局到 bottom 边。

Action

更换其他可用位置。

TA1001

WARN(TA1001): Either option "-name" or option "<source objects>" should be specified

sdc 约束 create_clock 既没有指定目标位置也没有指定时钟的名字。

```
create_clock -period 10 -waveform {0 5}
```

Action

修改 sdc 约束，增加时钟名称并且指定目标位置。

```
create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk}]
```

TA1004

WARN(TA1004): Clock waveform should not contain two edges with time larger than one clock period

sdc 约束 create_clock 约束中，-waveform 指定的时钟边沿大于-period 指定的周期。

```
create_clock -name clk1 -period 10 -waveform {0 15} [get_ports {clk}]
```

Action

修改 create_clock 约束中-waveform 或者-period 的大小，确保时钟边沿在一个时钟周期内。

```
create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk}]
```

TA1006

WARN(TA1006): The waveform list is not monotonically increasing

sdc 约束 create_clock 约束中，-waveform 指定的时钟边沿不是递增的。

```
create_clock -name clk1 -period 10 -waveform {5 0} [get_ports {clk}]
```

Action

修改 create_clock 约束中-waveform，确保指定的时钟边沿是递增的。

```
create_clock -name clk1 -period 10 -waveform {5 10} [get_ports {clk}]
```

TA1011

WARN(TA1011): Option "-rise" and option "-fall" are mutually exclusive

set_input_delay/set_output_delay 约束中，-rise -fall 同时使用。-rise 和 -fall 是互斥的，在同一条约束中只能使用一个。

```
create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk}]

set_input_delay -clock clk1 1 -rise –fall -max [get_ports {in01}]
```

Action

修改 set_input_delay/set_output_delay 约束，同一条约束中-rise 和-fall 只使用一个。

```
create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk}]

set_input_delay -clock clk1 1 -rise -max [get_ports {in01}]

set_input_delay -clock clk1 1 -fall -max [get_ports {in01}]
```

TA1012

WARN(TA1012): Option "-max" and option "-min" are mutually exclusive

set_input_delay/set_output_delay 约束中，-max-min 同时使用。-max 和 -min 是互斥的，在同一条约束中只能使用一个。

```
create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk}]

set_input_delay -clock clk1 1 -rise –max -min [get_ports {in01}]
```

Action

修改 set_input_delay/set_output_delay 约束，同一条约束中-max 和 -min 只使用一个。

```
create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk}]

set_input_delay -clock clk1 1 -rise -max [get_ports {in01}]

set_input_delay -clock clk1 1 -rise -min [get_ports {in01}]
```

TA1016

WARN(TA1016):Options "-edges -edge_shift" and options "-divide_by -multiply_by -duty_cycle -phase -offset" are mutually exclusive: specify either of the two ways

sdc 约束 create_generated_clock 中，有两种方式确定衍生时钟的频率和相位：第一种是通过-edges 和-edge_shift；第二种是通过-divide_by/ -multiply_by/ -duty_cycle/ -phase /-offset。但是这两种方法不能混用，如果混用就会报告这个信息。

```
create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk}]

create_generated_clock -name genClk -source [get_ports {clk}]
-master_clock clk1 -edges {1 3 5} -edge_shift {1 1 1} -divide_by 2  [get_pins
{reg0_0_Z/Q}]
```

Action

只用一种方式来确定衍生时钟的频率和相位。

```
create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk}]

create_generated_clock -name genClk -source [get_ports {clk}]
-master_clock clk1 -divide_by 2  [get_pins {reg0_0_Z/Q}]

create_generated_clock -name genClk2 -source [get_ports {clk}]
-master_clock clk1 -edges {1 3 5} -edge_shift {1 1 1}  [get_pins
{reg0_0_Z/Q}] –add
```

TA1019

WARN(TA1019): Option“-edges” must be in non-decreasing order

create_generated_clock 约束中，-edges 指定的参数不是递增的。

```
create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk}]

create_generated_clock -name genClk -source [get_ports {clk}]
-master_clock clk1 -edges {3 1 5} -edge_shift {1 1 1} [get_pins {reg0_0_Z/Q}]
```

Action

修改 create_generated_clock 约束，保证-edges 指定的参数是递增的。

```
    create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk}]

    create_generated_clock -name genClk -source [get_ports {clk}]
-master_clock clk1 -edges {1 3 5} -edge_shift {1 1 1} [get_pins {reg0_0_Z/Q}]
```

TA1027

WARN(TA1027): Missing required clock latency delay

set_clock_latency 约束中，没有指定延迟的值。

```
    create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk}]

    set_clock_latency -source -late -fall  [get_ports {clk}] -clock
[get_clocks {clk1}]
```

Action

set_clock_latency 约束中，指定延迟的值。

```
    create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk}]

    set_clock_latency -source -late -fall  [get_ports {clk}] -clock
[get_clocks {clk1}] 1
```

TA1032

WARN(TA1032): Option "-from" must be used with "get_clocks"

set_clock_uncertainty 约束中，-from 后面必须使用 get_clocks。当使用别的方式时会报告这个信息。

```
    create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk}]

    set_clock_uncertainty 1 -setup -from [get_ports {clk}] -to [get_clocks
{clk1}]
```

Action

set_clock_uncertainty 约束中，-from 后面使用 get_clocks。

```
    create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk}]

    set_clock_uncertainty 1 -setup -from [get_clocks {clk1}] -to [get_clocks
{clk1}]
```

TA1033

WARN(TA1033): Option "-to" must be used with "get_clocks"

set_clock_uncertainty 约束中，-to 后面必须使用 get_clocks。当使用别的方式时会报告这个信息。

```
create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk}]
set_clock_uncertainty 1 -setup -from [get_clocks {clk1}] -to [get_ports {clk}]
```

Action

set_clock_uncertainty 约束中，-to 后面使用 get_clocks。

```
create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk}]
set_clock_uncertainty 1 -setup -from [get_clocks {clk1}] -to [get_clocks {clk1}]
```

TA1048

WARN(TA1048): Existing clock <name> is overwritten

时钟被覆盖。原因是 sdc 命令中创建时钟的名字与已有的时钟重复。

```
create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk}]
create_clock -name clk1 -period 20 -waveform {0 5} [get_ports {clk}] –add
```

Action

修改 sdc 约束中时钟名称，确保不重复。

```
create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk}]
create_clock -name clk2 -period 20 -waveform {0 5} [get_ports {clk}] –add
```

TA1049

WARN(TA1049): Object<name>alreadyhas one clock applied on it, if you want one more, please use -add option

同一个位置加多个时钟，从第二个时钟开始约束中需要加-add。如果没有加-add，会报告出这个信息。

```
create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk}]
create_clock -name clk2 -period 20 -waveform {0 5} [get_ports {clk}]
```

Action

同一个位置从第二个时钟开始约束中需要加-add。

```
create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk}]

create_clock -name clk2 -period 20 -waveform {0 5} [get_ports {clk}] -add
```

TA1052

WARN(TA1052): Generated clock is ignored

sdc 约束 create_generated_clock 约束中，由于-name 指定的时钟名已被主时钟定义或者<object>没有指定时钟的入口等问题，导致 generated clcok 创建失败，就会报告这个信息。

```
create_ generated_clock -name clk1 -source [get_ports{clkin}]
-master_clock clk -edges{2 6 8} -edge_shift {6 5 3}
```

Action

修改 create_ generated_clock 约束语句确保符合要求。

```
create_ generated_clock -name clk1 -source [get_ports{clkin}]
-master_clock clk -edges{2 6 8} -edge_shift {6 5 3} [get_ports {a}]
```

TA1058

WARN(TA1058): Input ports list has output ports <name>

sdc 约束 set_input_delay 中，输入端口列表中有输出端口。set_input_delay 只能加到输入端口，如果加到输出端口就会报告这个信息。

```
create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk}]

set_input_delay -clock clk1 1 -min -fall [get_ports {out}]
```

Action

set_input_delay 只加到输入端口上。

```
create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk}]

set_input_delay -clock clk1 1 -min -fall [get_ports {in}]
```

TA1059

WARN(TA1059): Output ports list has input ports <name>

sdc 约束 set_output_delay 中，输出端口列表中有输入端口。set_output_delay 只能加到输出端口，如果加到输入端口就会报告这个信息。

```
create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk}]
set_output_delay -clock clk1 1 -min -fall [get_ports {in}]
```

Action

set_output_delay 只加到输出端口上。

```
create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk}]
set_output_delay -clock clk1 1 -min -fall [get_ports {out}]
```

TA1061

WARN(TA1061): Cannot find objects matching with <name>

找不到 sdc 约束中指定的对象。sdc 命令中指定的对象无法找到时会报告这个信息。

```
create_clock -name clk1 -period 10 -waveform {0 5} [get_ports
{port_no_exist}]
```

Action

修改 sdc 约束中指定的对象名称，使用正确的名称。

```
create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk}]
```

TA1068

WARN(TA1068):Previous IO timing constraints are overwritten

I/O 约束被覆盖。set_input_delay/set_output_delay 可以通过两组参数来指定约束范围：-max/-min 和-rise/-fall。如果没有指定-max/-min，那么默认会对 max 和 min 都分析，-rise/-fall 同理。如果后面的约束范围与前面的约束范围有重合就会报告这个信息。

```
create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk}]
set_input_delay -clock clk1 1 -max -fall [get_ports {in01}]
set_input_delay -clock clk1 2 [get_ports {in01}]
```

Action

建议-max/-min 和-rise/-fall 都明确指定，不要使用默认值。

```
create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk}]
set_input_delay -clock clk1 1 -max -fall [get_ports {in01}]
set_input_delay -clock clk1 2 -min -rise [get_ports {in01}]
```

TA1076

WARN(TA1076): Generated clock Source list has source object <object> which specified by option "-source", this generated clock will be ignored

sdc 约束 create_generated_clock 中，目标位置是主时钟来源位置。create_generated_clock 的目标位置不能与来源位置相同，否则会报告这个信息。

```
    create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk}]

    create_generated_clock -name genClk -source [get_ports {clk}]
-master_clock clk1 -divide_by 2 [get_ports {clk}]
```

Action

目标位置设置为主时钟来源位置以外的其他位置。

```
    create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk}]

    create_generated_clock -name genClk -source [get_ports {clk}]
-master_clock clk1 -divide_by 2 [get_pins {reg0_0_Z/Q}]
```

TA1083

WARN(TA1083): <name> is port ,should use “get_ports”

sdc 约束中当 -through 参数是一个 port 时，必须使用 get_ ports。如下案例中，set_multicycle_path 约束中，-through 参数是一个 port，使用 get_ nets，当使用别的方式时会报告这个信息。

```
    set_multicycle_path -from [get_clocks {clk}] –through [get_nets{clk}]
-to [get_clocks {genClk}] -hold
```

Action

-though 使用 get_ports 得到端口名。

```
    set_multicycle_path -from [get_clocks {clk}] –through [get_ports{clk}]
-to [get_clocks {genClk}] -hold
```

TA1109

WARN (TA1109): Invalid speed grade is specified"

sdc 中设置了一个错误的速度等级。GW1N 系列商业级速度等级是 5,6；工业级速度等级是 4,5；车规级速度等级是 4。GW2A 系列商业级速度等级是 7,8；工业级速度等级是 6,7；车规级速度等级是 6。如果指定的速度等级不符合以上的要求就会报告这个错误信息。

```
set_operating_conditions -grade c -model slow -speed 1
```

Action

确保器件类型（GW1N\GW2A）、温度等级（工业级\商业级\车规级）、速度等级符合要求。

```
set_operating_conditions -grade c -model slow -speed 5
```

TA1114

WARN(TA1114): Invalid access is specified

sdc 命令中访问方式不正确。出现这个信息可能的原因有三种：

1. set_clock_groups 约束中，-group 后访问方式不是 get_clocks 或 all_clocks；
2. 路径约束（set_false_path/set_max_delay /set_min_delay /set_multicycle_path）-rise_from /-fall_from/ -rise_to/ -rise_fall 后访问方式不是 get_clocks 或 all_clocks；
3. 报告约束（report_timing/ report_exceptions）后既有-from_clock 又有-from[get_clocks{}]或者既有-to_clock 又有-to[get_clocks{}]。

```
create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk}]

create_clock -name clk2 -period 10 -waveform {0 5} [get_ports {clk}] -add

set_clock_groups -exclusive -group [get_ports {clk}]

set_false_path -rise_from [get_ports {in1}] -fall_to [get_ports {out00}]

report_timing -setup -from_clock [get_clocks {clk1}] -to_clock [get_clocks {clk1}] -from [get_clocks {clk1}] -to [get_clocks {clk2}]
```

Action

避免以上三种情况，采用正确的访问方式。

```
create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk}]

create_clock -name clk2 -period 10 -waveform {0 5} [get_ports {clk}] -add

set_clock_groups -exclusive -group [get_clocks {clk1 clk2}]

set_false_path –rise_from [get_clocks {clk1}] –fall_to [get_clocks {clk2}]

report_timing -setup -from_clock [get_clocks {clk1}] -to_clock [get_clocks {clk1}]
```

TA1125

NOTE (TA1125): More than <num> critical paths are ignored because having large logic level

设计中处于时序路径上逻辑级数超过软件支持的最大逻辑级数。

Action

更改设计中时序路径上逻辑级数都小于等于 250。

TA2002

ERROR(TA2002): Cannot get clock with name <name>

如果使用的时钟在使用之前没有被正确创建，会报告这个错误。

```
   create_generated_clock -name genClk -source [get_ports {clk}]
-divide_by 2
```

Action

确保时钟被使用之前已经正确创建。

```
   create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk}]

   create_generated_clock -name genClk -source [get_ports {clk}]
-master_clock clk1 -divide_by 2
```


GOWIN高云
智慧逻辑 定制未来