Back to Top

A
Allow cross-site access to current domain at the same time in STORESTD, STORESTDX
Alternative separator character for multiple values in a single field in STORESTD, STORESTDX, CREATE DATA, CREATE DATA, CREATE DATA

Personal Ads Selling Motorcycles Sale

Personal Ads Selling Motorcycles Sale Lady

Sensitive Cameras Sensitive Cameras Sensitive Mics - Surfers are personally sensitive to cameras

Sensitive STRATOS Knife - Inserts Chat Sensitive Mics Sale

Back to Top

Sensitive Cameras Sensitive and contemplation For number already contents keyword Sensitive Cameras Sensitive Mics Sale

Sensitive Cameras Sensitive and contemplation For number already contents keyword Sensitive Cameras Sensitive Mics Sale

Sensitive Cameras Sensitive and contemplation For number already contents keyword Sensitive Cameras Sensitive Mics Sale

Sensitive Cameras Sensitive and contemplation of antiques and contemplation For number ... Sensitive Cameras Sensitive Mics Sale

Sensitive Cameras Sensitive and contemplation Antiques and contemplation For number ... Sensitive Cameras Sensitive Mics Sale

Sensitive and contemplation For number already contents sensitive Antiques and contemplation ... Sensitive Cameras Sensitive Mics Sale

Sensitive and contemplation For number contents

CONSIST OF ALL 12 STATEMENTS Statement and Conclusion

STATE PRIMARY SUBORDINATE Statement - Primitive statement No. Statement - Primitive statement No. Statement - Primitive statement No.

VHDL Wait Statement error at <location>: condition clause with UNTIL keyword cannot contain contain 'EVENT predefined attribute

VHDL Wait Statement error at <location>: Wait Statement must contain condition clause with UNTIL keyword

VHDL warning at <location>: ENUM_ENCODING attribute specifies too many encoding values for enumeration type <type>

VHDL warning at <location>: Exemplar attribute or directive <name> is not supported in MAX+PLUS II Advanced Synthesis software version 2.1 and earlier -- only translate_off and translate_on are supported

VHDL warning at <location>: ignored assignment of value to null range

VHDL warning at <location>: ignored choice that is null range

VHDL warning at <location>: ignored choice with illegal range bounds

VHDL warning at <location>: ignored choice with meta-value <name> for synthesis

VHDL warning at <location>: ignored choice with meta-values

VHDL warning at <location>: ignored pragma <name> for constant net

VHDL warning at <location>: ignored synthesis directive attribute <name> because number of bits in <type> type does not match number of enumeration values in Type Declaration

VHDL warning at <location>: ignored unknown or unsupported pragma <name>

VHDL warning at <location>: ignored VHDL standard library NOW function, which is not supported for synthesis

VHDL warning at <location>: used initial value for variable <name> because it was never assigned

VHDL warning at <location>: used X for unrecognized character <name> in enumerated type

Virtual pin clock <name> for virtual pin <name> does not exist in the design -- auto-selecting clock

Virtual pin clock not found for <number> virtual pins -- used Auto clock

Virtual pins cannot be assigned to ports requiring real pins

Back to Top

W

waddr port is not connected in WYSIWYG RAM primitive <name>

When in its current operating mode, WYSIWYG RAM primitive <name> cannot have <name> parameter

WYSIWYG CAM primitive <name> cannot have matchfound port connected when in an unencoded mode or a multiple match mode

WYSIWYG CAM primitive <name> has actual ADDRESS_WIDTH <number>, which is different from declared ADDRESS_WIDTH <number>

WYSIWYG CAM primitive <name> has actual literal width <number>, which is different from literal width <number> declared in parameter

WYSIWYG CAM primitive <name> must have a datain port

WYSIWYG CAM primitive <name> must have a lit port

WYSIWYG CAM primitive <name> must have a waddr port

WYSIWYG CAM primitive <name> must have a we port

WYSIWYG CAM primitive <name> must have a wrinvert port

WYSIWYG CAM primitive <name> must have content-addressable memory ADDRESS_WIDTH parameter

WYSIWYG CAM primitive <name> must have LOGICAL_CAM_DEPTH parameter

WYSIWYG CAM primitive <name> must have LOGICAL_CAM_WIDTH parameter

WYSIWYG CAM primitive <name> must have outputselect port connected only when OPERATION_MODE parameter is set to UNENCODED_32_ADDRESS or MULTIPLE_MATCH

WYSIWYG CAM primitive <name> must have outputselect port connected when the OPERATION_MODE parameter is set to UNENCODED_32_ADDRESS or MULTIPLE_MATCH

WYSIWYG ClockLock PLL primitive <name> cannot be fed by logic

WYSIWYG CRC block or remote update block primitive <name> must use <name> port

WYSIWYG I/O <name> operation mode is <type>, which does not match padio pin <name> type of <type>

WYSIWYG I/O operation modes do not match their padio pin's type

WYSIWYG I/O primitive <name> cannot be converted to an APEX II WYSIWYG I/O primitive

WYSIWYG I/O primitive <name> cannot have both a clear signal and power-up high, or both a preset signal and power-up low

WYSIWYG I/O primitive <name> cannot have both aclr port and a POWER_UP parameter value set to HIGH

WYSIWYG I/O primitive <name> cannot have extra register control signals

WYSIWYG I/O primitive <name> cannot have OPEN_DRAIN_OUTPUT parameter value set to TRUE when its I/O element is in bidirectional operation mode and uses an output enable port

WYSIWYG I/O primitive <name> cannot specify modes related to unused register

WYSIWYG I/O primitive <name> converted to equivalent logic

WYSIWYG I/O primitive <name> has illegal OPERATION_MODE parameter value <name>

WYSIWYG I/O primitive <name> has illegal value for FEEDBACK_MODE parameter

WYSIWYG I/O primitive <name> has illegal value for OPEN_DRAIN_OUTPUT parameter

WYSIWYG I/O primitive <name> has illegal value for REG_SOURCE_MODE parameter

WYSIWYG I/O primitive <name> has OE_REGISTER_MODE parameter value set to REGISTER, but does not use oe port

WYSIWYG I/O primitive <name> has OPERATION_MODE parameter value set to BIDIR, but does not have oe port

WYSIWYG I/O primitive <name> is dependent on unconnected register control signals

WYSIWYG I/O primitive <name> is in unsupported mode

WYSIWYG I/O primitive <name> must have padio port

WYSIWYG I/O primitive <name> uses its clk port, but does not use its register dataout port

WYSIWYG I/O primitive <name>, which has OPERATION_MODE parameter value set to either OUTPUT or BIDIR, must have data source

WYSIWYG I/O primitives converted to equivalent logic

WYSIWYG I/Os found with illegal padio connections

WYSIWYG LCELL primitive <name> cannot have cin port when its OPERATION_MODE parameter is set to QFBK_COUNTER

WYSIWYG LCELL primitive <name> cannot have cout port when in normal mode

WYSIWYG LCELL primitive <name> cannot have improperly connected cascin port

WYSIWYG LCELL primitive <name> cannot have PACKED_MODE parameter set to TRUE when not in normal mode

WYSIWYG LCELL primitive <name> cannot have sload and/or sclr port(s) when in non-counter modes

WYSIWYG LCELL primitive <name> cannot have sload and/or sclr port(s) when it also has combout port or is in non-arithmetic modes

WYSIWYG LCELL primitive <name> cannot use both datad and cout ports when in counter mode

WYSIWYG LCELL primitive <name> cin0, cin1, cout0, and cout1 ports cannot be used in design entry

WYSIWYG LCELL primitive <name> fans out via carry or cascade chains to more than one destination WYSIWYG LCELL primitive

WYSIWYG LCELL primitive <name> feeds more than one logic cell via carry or cascade chains

WYSIWYG LCELL primitive <name> has cascin port that cannot be inverted, VCC, or GND

WYSIWYG LCELL primitive <name> has cin and cascin ports connected to different WYSIWYG LCELL primitives, but the ports must be connected to the same WYSIWYG LCELL primitive

WYSIWYG LCELL primitive <name> has cin port that cannot be inverted, VCC, or GND

WYSIWYG LCELL primitive <name> has cout or cascout port that feeds another WYSIWYG LCELL primitive's input port other than cin or cascin port

WYSIWYG LCELL primitive <name> has illegal cout port with fan-out when in normal mode

WYSIWYG LCELL primitive <name> has illegal value for LUT_MASK parameter

WYSIWYG LCELL primitive <name> has illegal value for PACKED_MODE parameter

WYSIWYG LCELL primitive <name> has LUT_MASK parameter that is not set to VCC or GND, but has no data inputs

WYSIWYG LCELL primitive <name> has PACKED_MODE parameter set to TRUE, but uses cin or cout port

WYSIWYG LCELL primitive <name> has PACKED_MODE parameter value set to TRUE, but does not use datac port

WYSIWYG LCELL primitive <name> has PACKED_MODE parameter value set to TRUE, but does not use datad port

WYSIWYG LCELL primitive <name> has synchronous control signals that are not part of a carry chain.

WYSIWYG LCELL primitive <name> is dependent on unconnected inputs

WYSIWYG LCELL primitive <name> is operating in counter mode, but this mode no longer exists for the current device family -- setting the operating mode to arithmetic

WYSIWYG LCELL primitive <name> must have LUT_MASK parameter

WYSIWYG LCELL primitive <name> must not use both regout and combout ports

WYSIWYG LCELL primitive <name> must not use combout port when in counter mode

WYSIWYG LCELL primitive <name> must not use datac port when in arithmetic mode

WYSIWYG LCELL primitive <name> must not use datad port when in arithmetic mode

WYSIWYG LCELL primitive <name> must use an adata port when aload port is used

WYSIWYG LCELL primitive <name> must use sdata port when sload port is used

WYSIWYG LCELL primitives have synchronous control signals that are not part of a carry chain

WYSIWYG LVDS receiver primitive <name> has illegal value for CHANNEL_WIDTH parameter

WYSIWYG LVDS transmitter primitive <name> has illegal CHANNEL_WIDTH parameter

WYSIWYG LVDS transmitter primitive <name> has value for CHANNEL_WIDTH parameter that does not correspond to the number of datain port connections

WYSIWYG MCELL primitive <name> cannot use both aclr and paclr ports

WYSIWYG MCELL primitive <name> cannot use both clk and pclk ports

WYSIWYG MCELL primitive <name> has illegal combination of mode parameter settings -- cannot be in VCC mode and combinatorial mode when parallel expander mode is turned off

WYSIWYG MCELL primitive <name> has illegal combination of mode parameters -- cannot be in XOR or XNOR mode, TFF mode, and parallel expander mode

WYSIWYG MCELL primitive <name> has illegal value for PEXP_MODE parameter -- value must be ON or OFF

WYSIWYG MCELL primitive <name> has illegal value for POWER_UP parameter -- value must be HIGH or LOW

WYSIWYG MCELL primitive <name> has illegal value for REGISTER_MODE parameter -- value must be DFF or TFF

WYSIWYG MCELL primitive <name> register must use fpin fast input port if it is in register mode, DFF mode, and VCC mode

WYSIWYG primitive <name> belongs to wrong device family

WYSIWYG primitive <name> cannot have <name> parameter set to <value> <text>

WYSIWYG primitive <name> cannot have BIT_NUMBER parameter value greater than or equal to LOGICAL_RAM_WIDTH value

WYSIWYG primitive <name> cannot have clk port when regout port is not used

WYSIWYG primitive <name> cannot have datain port when in read-only mode

WYSIWYG primitive <name> cannot have FIRST_PATTERN_BIT parameter value greater than or equal to LOGICAL_CAM_WIDTH parameter value

WYSIWYG primitive <name> cannot have pexpout port when its OPERATION_MODE parameter is not set to PTERM_EXP or PACKED_PTERM_EXP

WYSIWYG primitive <name> cannot have re port when in read-only mode

WYSIWYG primitive <name> cannot have re port when in single-port mode

WYSIWYG primitive <name> cannot have READ_ADDRESS_CLEAR parameter when in deep RAM mode

WYSIWYG primitive <name> cannot have waddr port when in read-only mode

WYSIWYG primitive <name> cannot have we port when in read-only mode

WYSIWYG primitive <name> cannot use clk port when its OPERATION_MODE parameter is set to PTERM_EXP

WYSIWYG primitive <name> cannot use clk port when its OUTPUT_MODE parameter is set to COMB

WYSIWYG primitive <name> cannot use dataout port when OPERATION_MODE parameter is set to PTERM_EXP

WYSIWYG primitive <name> cannot use pterm5 port when in XOR mode and when pexpout port is not used

WYSIWYG primitive <name> cannot use registered dataout port when in deep RAM mode

WYSIWYG primitive <name> clock port must be fed by input pin, and must not be fed by any logic gates

WYSIWYG primitive <name> does not have OUTPUT_MODE parameter value specified and is defaulting to COMB

WYSIWYG primitive <name> does not have OUTPUT_MODE parameter value specified and is defaulting to REG

WYSIWYG primitive <name> has <name> port that must be connected <text>

WYSIWYG primitive <name> has <name> port that must not be connected <text>

WYSIWYG primitive <name> has <name> port that should be connected <text>

WYSIWYG primitive <name> has <name> port with a specified width of <number>, but has <number> more bits connected than the specified width

WYSIWYG primitive <name> has <name> port with specified width of <number>, but only <number> bits are connected

WYSIWYG primitive <name> has aclr port, but does not have regout port

WYSIWYG primitive <name> has aclr port, but does not use its register

WYSIWYG primitive <name> has aload port, but does not have regout port

WYSIWYG primitive <name> has aload port, but does not use its register

WYSIWYG primitive <name> has apre port, but does not have regout port

WYSIWYG primitive <name> has apre port, but does not use its register

WYSIWYG primitive <name> has clk port, but does not contain regout port

WYSIWYG primitive <name> has CLOCK_ENABLE_MODE parameter value that is set to TRUE, but does not use its dataa port

WYSIWYG primitive <name> has ena port, but does not have regout port

WYSIWYG primitive <name> has ena port, but does not use its register

WYSIWYG primitive <name> has ena0 port, but does not have clk0 port

WYSIWYG primitive <name> has ena1 port, but does not have clk1 port

WYSIWYG primitive <name> has illegal pexpin port -- cannot be VCC, GND, inverted, direct from pin, or from non-parallel expander output port on another WYSIWYG primitive

WYSIWYG primitive <name> has illegal value for <name> parameter

WYSIWYG primitive <name> has illegal value for <name> parameter -- value must be a number

WYSIWYG primitive <name> has illegal value for DATA_IN_CLEAR parameter

WYSIWYG primitive <name> has illegal value for DATA_IN_CLOCK parameter

WYSIWYG primitive <name> has illegal value for DATA_OUT_CLEAR parameter

WYSIWYG primitive <name> has illegal value for DATA_OUT_CLOCK parameter

WYSIWYG primitive <name> has illegal value for INVERT_PTERM1_MODE parameter

WYSIWYG primitive <name> has illegal value for OPERATION_MODE parameter

WYSIWYG primitive <name> has illegal value for OUTPUT_CLEAR parameter

WYSIWYG primitive <name> has illegal value for OUTPUT_CLOCK parameter

WYSIWYG primitive <name> has illegal value for OUTPUT_MODE parameter

WYSIWYG primitive <name> has illegal value for PATTERN_WIDTH parameter

WYSIWYG primitive <name> has illegal value for READ_ADDRESS_CLEAR parameter

WYSIWYG primitive <name> has illegal value for READ_ADDRESS_CLOCK parameter

WYSIWYG primitive <name> has illegal value for READ_ENABLE_CLEAR parameter

WYSIWYG primitive <name> has illegal value for READ_ENABLE_CLOCK parameter

WYSIWYG primitive <name> has illegal value for WRITE_ADDRESS_CLEAR parameter

WYSIWYG primitive <name> has illegal value for WRITE_ENABLE_CLEAR parameter

WYSIWYG primitive <name> has illegal value for WRITE_LOGIC_CLEAR parameter

WYSIWYG primitive <name> has illegal value for WRITE_LOGIC_CLOCK parameter

WYSIWYG primitive <name> has incompatible parameters

WYSIWYG primitive <name> has inconsistent parameter values <name> and <name>

WYSIWYG primitive <name> has incorrect Hexadecimal (Intel-Format) File name <name> for LPM_FILEX or INIT_FILEX parameter

WYSIWYG primitive <name> has sclr port, but does not use its register

WYSIWYG primitive <name> has sload port, but does not use its register

WYSIWYG primitive <name> is in unsupported dual-port mode -- must have waddr and wena ports that are both either registered or non-registered

WYSIWYG primitive <name> is in unsupported ROM mode -- must have ports that are both either registered or non-registered

WYSIWYG primitive <name> is in unsupported single-port mode -- must have waddr and raddr ports that are both either registered or non-registered

WYSIWYG primitive <name> is missing required <name> parameter

WYSIWYG primitive <name> is missing value for <name> parameter. Legal values for <name> parameter are <values>.

WYSIWYG primitive <name> must have <name> parameter set to <value> <text>

WYSIWYG primitive <name> must have BIT_NUMBER parameter

WYSIWYG primitive <name> must have clk port

WYSIWYG primitive <name> must have clk port when OPERATION_MODE parameter is set to TFF or TBARFF

WYSIWYG primitive <name> must have clk port when OUTPUT_MODE parameter is set to REG

WYSIWYG primitive <name> must have clk0 port

WYSIWYG primitive <name> must have clk1 port

WYSIWYG primitive <name> must have CLOCK_ENABLE_MODE parameter

WYSIWYG primitive <name> must have clr0 port

WYSIWYG primitive <name> must have clr1 port

WYSIWYG primitive <name> must have DATA_OUT_CLOCK parameter when value is specified for its DATA_OUT_CLEAR parameter

WYSIWYG primitive <name> must have FIRST_ADDRESS parameter

WYSIWYG primitive <name> must have FIRST_PATTERN_BIT parameter

WYSIWYG primitive <name> must have LAST_ADDRESS parameter

WYSIWYG primitive <name> must have legal value for ADDRESS_WIDTH parameter

WYSIWYG primitive <name> must have LOGICAL_CAM_NAME parameter

WYSIWYG primitive <name> must have LOGICAL_RAM_WIDTH parameter

WYSIWYG primitive <name> must have OPERATION_MODE parameter

WYSIWYG primitive <name> must have PATTERN_WIDTH parameter

WYSIWYG primitive <name> must have pterm0 port when in packed product term expander mode

WYSIWYG primitive <name> must have READ_ADDRESS_CLEAR parameter when value is specified for its READ_ADDRESS_CLOCK parameter

WYSIWYG primitive <name> must have SUM_LUTC_INPUT parameter set to CIN if cin port is used when in normal mode

WYSIWYG primitive <name> must not use pxor port in non-XOR mode

WYSIWYG primitive <name> must not use regcascin port when not in register cascade mode

WYSIWYG primitive <name> must use all pterm (pterm0 - pterm4) ports when in XOR mode and when pexpout port is not used

WYSIWYG primitive <name> must use cin port if SUM_LUTC_INPUT is set to CIN

WYSIWYG primitive <name> must use clk port if SUM_LUTC_INPUT parameter is set to QFBK

WYSIWYG primitive <name> must use clk0 port

WYSIWYG primitive <name> must use clk1 port if ena1 port is used

WYSIWYG primitive <name> must use either clk or pclk port for registered output

WYSIWYG primitive <name> must use pexpin port or at least one pterm port if pexpout port is used

WYSIWYG primitive <name> must use pterm5 port when in XOR mode and in DFF mode, and when using the pexpout port

WYSIWYG primitive <name> must use pxor port when in XOR mode

WYSIWYG primitive <name> must use regcascin port when in register cascade mode

WYSIWYG primitive <name> padio port connected with ATOM <name> must connect to only a single top-level pin of the same type, but feeds other logic

WYSIWYG primitive <name> padio port is connected illegally

WYSIWYG primitive <name> parameter value is out of range

WYSIWYG primitive <name> sclr or sload port must be used when in synchronous mode

WYSIWYG primitive <name> sum LUT mask must depend on datac input signal value if SUM_LUTC_INPUT parameter is set to QFBK

WYSIWYG primitive <name> uses fpin fast input port that does not feed register port -- it must feed register port. In addition, WYSIWYG primitive must not use pterm5 and pxor ports, and, if PEXP_MODE parameter is set to OFF, it must not use the pterm0 - pterm4 ports.

WYSIWYG primitive <name> uses pexpin port, but must use all five pterm (pterm0 - pterm4) ports

WYSIWYG primitive <name> uses pexpout port that must feed pexpin port

WYSIWYG primitive <name> uses portabyteenamasks port, but PORT_A_DATA_WIDTH parameter is not a byte multiple of width of portabyteenamasks port, and portadatain port is not used

WYSIWYG primitive <name> uses portbbyteenamasks port, but PORT_B_DATA_WIDTH parameter is not a byte multiple of width of portbbyteenamasks port, and portbdatain port is not used

WYSIWYG primitive <name> uses pterm5 port, but pterm5 port is not supported for non-parallel expander or VCC modes

WYSIWYG primitive <WYSIWYG name> will now power up <high or low>

WYSIWYG RAM primitive <name> cannot have <names> ports in mixed registered modes -- all ports must be registered or non-registered

WYSIWYG RAM primitive <name> has <names> parameters with values that do not match between read and write addresses for <A or B> port

WYSIWYG RAM primitive <name> has actual ADDRESS_WIDTH <number>, which is different from declared ADDRESS_WIDTH value <number>

WYSIWYG RAM primitive <name> has actual DATA_WIDTH <number>, which is different from declared DATA_WIDTH <number>

WYSIWYG RAM primitive <name> has ADDRESS_WIDTH <number>, which is incompatible with LOGICAL_RAM_DEPTH value <number> declared in parameter

WYSIWYG RAM primitive <name> has different clear signals feeding bits of <name> input bus port

WYSIWYG RAM primitive <name> has different clock signals feeding bits of <name> input bus port

WYSIWYG RAM primitive <name> has mismatched <names> parameters for <name> port and <name> port

WYSIWYG RAM primitive <name> has RAM ADDRESS_WIDTH parameter value that must be in deep RAM mode

WYSIWYG RAM primitive <name> has too many bits

WYSIWYG RAM primitive <name> must feed a single tri-state bus node

WYSIWYG RAM primitive <name> must have <name> parameter

WYSIWYG RAM primitive <name> must have <name> port or parameter specified

WYSIWYG RAM primitive <name> must have LOGICAL_RAM_DEPTH parameter

WYSIWYG RAM primitive <name> must have RAM ADDRESS_WIDTH parameter

WYSIWYG RAM primitive <name> must have same values specified for both its WRITE_ENABLE_CLEAR and WRITE_ADDRESS_CLEAR parameters

WYSIWYG RAM primitive <name> uses <name> port, which is inconsistent with <name> parameter value

Back to Top

X

Back to Top

Y

Back to Top

Z

Back to Top

aload port and apre port cannot be connected at the same time in WYSIWYG LCELL primitive <*name*>

CAUSE: The specified WYSIWYG LCELL primitive has both an `aload` port and an `apre` port that are connected at the same time. One of the two must be disconnected from the WYSIWYG primitive, or the WYSIWYG primitive must be changed to emulate `apre` and `aload`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

Assignment <*name*>=<*name*> on location <*name*> cannot be moved to location <*name*> because it conflicts with existing assignment <*name*>=<*name*>

CAUSE: The specified assignment on the specified location cannot be moved to the specified location because the specified assignment conflicts with the specified existing assignment. As a result, the name of the logic on the first specified assignment will be lost. To keep the name of the logic on the first specified assignment, the MAX+PLUS II Advanced Synthesis software will move the assignments to another location that shares the same net as the logic that is being removed.

ACTION: Remove or change one of the conflicting assignments.

BIDIR pin <*name*> does not have a source

CAUSE: You attempted to compile a design. However, the MAX+PLUS II Advanced Synthesis software stopped compiling because the design contains the specified BIDIR pin that is not driven by a source.

ACTION: Either add a source to the specified BIDIR pin or convert it to an INPUT pin.

Black-boxed entity <*name*>: <*name*>

CAUSE: You instantiated the specified entity in the design in a language other than Verilog/VHDL. The Compiler detected the specified entity and has created a synthesis black box for it.

ACTION: No action is required.

Black-boxed design unit <*name*>: <*name*>

CAUSE: You instantiated the specified entity in the design in a language other than Verilog/VHDL. The Compiler detected the specified entity and has created a synthesis black box for it.

ACTION: No action is required.

Black-boxed <*number*> design units and <*number*> entities in source file <*name*>

CAUSE: You instantiated the specified number of design units in the design, and the Compiler has identified the specified number of entities in the specified source file. These design units and entities are not written in Verilog or VHDL, and so the Compiler has created synthesis black boxes for them.

ACTION: No action is required.

Block type <*name*> for WYSIWYG RAM primitive <*name*> is not supported in target device family

---

CAUSE: The specified block type for the specified WYSIWYG RAM primitive is not supported in the target device family.

ACTION: Check the design to make sure the proper memory block type for the target device family is specified. If the problem still exists, contact Altera Applications for assistance.

Can't analyze file -- file <*name*> is missing

CAUSE: You attempted to compile the current design, but the Database Builder cannot find the specified file that is needed for compilation. This message can occur if the specified file was deleted or renamed or if the directory name or path name does not exist.

ACTION: Verify that the specified file exists. If the file exists, create the directory for the file or specify the correct path for the file and then recompile the design.

Can't choose best encoding method for state machine *<name>*

CAUSE: You set the **State Machine Processing** logic option to **Auto**; however, the logic of the specified state machine is too complex for the Logic Synthesizer to choose the best state machine encoding method.

ACTION: No action is required. The Logic Synthesizer attempts to process the logic using either one-hot encoding or simple binary encoding. To avoid receiving this message in the future, you might try simplifying the logic of the state machine.

Can't connect TRI or OPNDRN primitive <*name*> to pin <*name*> because it is already connected to bidirectional pin <*name*>

CAUSE: The specified `TRI` or `OPNDRN` primitive is connected to the two specified pins, one of which is bidirectional. However, the `TRI` or `OPNDRN` primitives must be separated from other pins by logic. If the `TRI` or `OPNDRN` primitives drive a bidirectional pin, the primitive can drive other primitives, but not other pins.

ACTION: Modify the design so that the `TRI` or `OPNDRN` primitives do not drive a bidirectional pin and another pin simultaneously.

Can't continue processing -- expected file <*name*> is missing

---

CAUSE: You performed one of the following actions:

- Analyzed a file
- Created an AHDL Include File (**.inc**)
- Created a Block Symbol File (**.bsf**)
- Created a VHDL Design File (**.vhd**) or Verilog Design File (**.v**) for the current Block Design File (**.bdf**).

However, the MAX+PLUS II Advanced Synthesis software cannot continue the process because the specified file that is required for the process is missing.

ACTION: Verify that the specified file exists. If the file exists, create the directory for the file or specify the correct path for the file and then restart the process.

Can't convert CLKLOCK primitives to ClockLock PLLs -- ClockLock PLLs not supported in target device family

---

CAUSE: The design contains `CLKLOCK` primitives that could not be converted to ClockLock PLLs because the current target device you specified for the design does not support ClockLock PLLs.

ACTION: Remove the `CLKLOCK` primitives from the design, or target a device family that supports ClockLock PLLs.

Can't create database directory for project in project directory *<name>*

CAUSE: The MAX+PLUS II Advanced Synthesis software cannot create a database directory for the project in the current project directory. This message can occur when you specify an illegal directory or path name, you do not have permission to write to the specified directory, a file or directory with the same name already exists, or there is a disk error, for example, you do not have enough available drive space.

ACTION: Make sure you have permission to write to the directory, and check the hard disk for any errors and available space.

Can't create symbol for entity <*name*> -- declaration for port <*name*> cannot be of complex type

---

CAUSE: You tried to create a symbol for a VHDL Design File (**.vhd**). However, the Entity Declaration in the VHDL Design File contains ports with types that are too complex (for example, ports with integer types or unconstrained array types). As a result, the MAX+PLUS II Advanced Synthesis software cannot create a symbol from the VHDL Design File.

ACTION: Change the types of the ports in the VHDL Design File to simple types, such as bit types or one-dimensional array types.

See also:

Creating a Block Symbol File for a Current File
Sections 1.1 and 3 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

Can't create symbol or include file -- file <*name*> is a BDF or compatible file

---

CAUSE: You attempted to create a Block Symbol File (**.bsf**) or an AHDL Include File (**.inc**) for the current file. However, the MAX+PLUS II Advanced Synthesis software cannot create the file because the specified file that is required for this process cannot be a BDF or a compatible file.

ACTION: Create a BSF or INC File that supports BDF or compatible files.

Can't create VHDL Design File or Verilog Design File -- file <*name*> is not a Block Design File or compatible file

---

CAUSE: You attempted to create a VHDL Design File (**.vhd**) or Verilog Design File (**.v**) for the current Block Design File (**.bdf**). However, the MAX+PLUS II Advanced Synthesis software cannot create the file because the specified file that is required for this process is not a BDF or a compatible file.

ACTION: Verify that the specified file exists. If the file exists, create the directory for the file or specify the correct path for the file and then restart the process.

Can't feed gate <*name*> of type <*type*> with CASCADE primitive <*name*>

CAUSE: The specified `CASCADE` primitive feeds the specified gate primitive, but a `CASCADE` primitive can only feed `AND` or `OR` gate primitives, or `DFF` primitives.

ACTION: Modify the design so the `CASCADE` primitive feeds an `AND` or `OR` gate primitive, or a `DFF` primitive.

Can't feed gate primitive <*name*> with CASCADE primitive <*name*> because it is already fed by CASCADE primitive <*name*>

CAUSE: The specified gate primitive is fed by the specified `CASCADE` primitive, but is already fed by the another specified `CASCADE` primitive. A `CASCADE` primitive can feed only by a single gate, which must be an `AND` or an `OR` gate primitive.

ACTION: Modify the design by removing one of the conflicting `CASCADE` primitives, so that the gate primitive is fed by only one `CASCADE` primitive.

Can't feed logic gate <*name*> of type <*type*> by tri-state signal

---

CAUSE: You connected the specified logic gate to a tri-state signal (Z logic value), but you cannot create internal tri-state buffers. Therefore, if the signal feeding into the node is Z, it will become a "1" instead.

ACTION: No action is required.

Can't feed NOT gate <*type*> <*name*> by tri-state signal

CAUSE: You connected the specified `NOT` gate to a tri-state signal (Z value), but you cannot create internal tri-state buffers. Therefore, if the signal feeding in is Z, it will become a "1" instead.

ACTION: No action is required.

Can't feed XOR gate <*name*> with CASCADE primitive <*name*>

CAUSE: The specified `CASCADE` primitive feeds an `XOR` gate, however, a `CASCADE` primitive can only feed a single gate primitive, which must be an `AND` or an `OR` gate primitive..

ACTION: Modify the design by removing the `CASCADE` primitive, or modify logic in the design so the `CASCADE` primitive feeds an `AND` or `OR` gate.

Can't find auto-select clock for virtual pin <*name*> -- setting clock to GND

CAUSE: The Compiler attempted to automatically assign the **Virtual Pin Clock** logic option to an appropriate clock signal in the design for the specified **Virtual Pin**. However, the Compiler cannot locate an appropriate clock signal for this purpose. When this condition occurs, the Compiler sets the virtual pin clock to `GND`.

ACTION: Assign the **Virtual Clock Pin** logic option to an appropriate clock signal in the design.

Can't find corresponding PADIO pin for WYSIWYG I/O primitive *<name>*

CAUSE: The specified WYSIWYG I/O primitive does not correspond to pins that connect to its `padio` ports. As a result, the MAX+PLUS II Advanced Synthesis software cannot find the corresponding pin for the specified primitive.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance. This message is a submessage of the message that precedes it in the Messages window and in the Messages section of the Report window.

Can't find corresponding PADIO pins for the following WYSIWYG I/O primitives

---

CAUSE: The WYSIWYG I/O primitives listed in the submessage(s) of this message do not correspond to pins that connect to their `padio` ports.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

Can't find database file <*name*>

CAUSE: The MAX+PLUS II Advanced Synthesis software cannot locate the specified database file, or the specified directory name or path name does not exist. This message can occur if the specified database file or directory was deleted or moved outside the MAX+PLUS II Advanced Synthesis software by another user.

ACTION: Verify that the specified database file and specified directory exist. If the directory does not exist, create the directory for the file or specify the correct path for the file and then recompile the design. Deleting the database file erases the results of the previous compilation; recompiling the design restores the database file.

Can't find license for core <*name*>

CAUSE: You instantiated the specified core and attempted to compile the design. However, the MAX+PLUS II Advanced Synthesis software cannot find a license for the specified core. As a result, the MAX+PLUS II Advanced Synthesis software will compile the design, but will not generate a programming file or any simulation netlist files for the core.

ACTION: Contact Altera Applications for assistance with purchasing a full license for the core.

Can't implement CASCADE primitive <*name*> -- polarity of CASCADE primitives and other primitives in cascade chain are inconsistent

---

CAUSE: The MAX+PLUS II Advanced Synthesis software cannot implement the specified `CASCADE` primitive because the polarity in a cascade chain must be the same for all primitives in the chain.

ACTION: Modify the design by inverting the fan-out for the `CASCADE` primitive or remove the `CASCADE` primitive.

Can't implement in ROM

CAUSE: You used the **Assignment Organizer** command (Assignments menu) to turn on the **Auto Implement in ROM** option for a particular entity, but the Compiler could not implement the entity in ROM.

ACTION: Refer to additional warning message(s) in the Messages window to determine the specific reasons the Compiler cannot implement the entity in ROM.

Can't implement in ROM -- DFF primitive <*name*> is a buried register or contains conflicting control signals

---

CAUSE: You used the **Assignment Organizer** command (Assignments menu) to turn on the **Auto Implement in ROM** option for the specified `DFF` primitive, but the Compiler cannot implement the `DFF` in ROM because it is a buried register or contains conflicting control signals.

ACTION: Make sure the `DFF` primitive is fed by a pin or feeds a pin and all control signals of the input or output registers match.

Can't implement in ROM -- hierarchy has <*number*> inputs

CAUSE: You used the **Assignment Organizer** command (Assignments menu) to turn on the **Auto Implement in ROM** option for a particular entity. However, the Compiler could not implement the entity in ROM because the current hierarchy contains too many inputs.

ACTION: Turn off the **Auto Implement in ROM** option.

Can't implement in ROM -- hierarchy is too simple

CAUSE: You used the **Assignment Organizer** command (Assignments menu) to turn on the **Auto Implement in ROM** option, but the hierarchy is too simple to implement in ROM. For example, you may have turned on the **Auto Implement in ROM** option for logic that reduces to a single node.

ACTION: Turn off the **Auto Implement in ROM** option.

Can't implement in ROM -- input pin <*name*> feeds more than one DFF primitive

---

CAUSE: You used the **Assignment Organizer** command (Assignments menu) to turn on the **Auto Implement in ROM** option for the specified input pin. However, the Compiler cannot implement the pin in ROM because the pin feeds more than one `DFF` primitive.

ACTION: If the input pin feeds the `D_IN` port of a `DFF` primitive, make sure the pin has only one fan-out, or turn off the **Auto Implement in ROM** option.

Can't implement in ROM -- node <*name*> of type <*type*> is not supported

---

CAUSE: You used the **Assignment Organizer** command (Assignments menu) to turn on the **Auto Implement in ROM** option for the specified node. However, the Compiler could not implement the node in ROM because the node is not supported.

ACTION: Turn off the **Auto Implement in ROM** option for the node.

Can't implement in ROM -- output pin <*name*> is fed by VCC or GND

CAUSE: You used the **Assignment Organizer** command (Assignments menu) to turn on the **Auto Implement in ROM** option for the specified input pin. However, the Compiler cannot implement the pin in ROM because the pin is fed by `VCC` or `GND`.

ACTION: Modify the design so that the output pin is not fed by `VCC` or `GND`, or turn off the **Auto Implement in ROM** option for the output pin.

Can't implement option(s) assigned to entity <*name*> due to bidirectional connections between hierarchies

---

CAUSE: You assigned an entity-specific logic option to the specified entity. Implementing this entity-specific logic option requires hierarchical synthesis. However, the bidirectional connection to the top-level entity prevents hierarchical synthesis. As a result, the MAX+PLUS II Advanced Synthesis software cannot implement the assignment.

ACTION: Remove the entity-specific logic option or place the tri-state bus entirely within a single entity.

Can't implement option(s) assigned to entity <*name*> due to netlist optimizations using Fitter timing information

---

CAUSE: You attempted to optimize netlists by directing the MAX+PLUS II Advanced Synthesis software to use timing information from the Fitter to improve the quality of synthesis. However, this option is incompatible with one or more entity-specific logic options assigned to the specified entity because the netlist optimization process flattens the netlist. As a result, the MAX+PLUS II Advanced Synthesis software cannot implement the entity-specific logic option assignment.

ACTION: If you want to use Fitter timing information for netlist optimization, remove the entity-specific logic option. Otherwise, specify that the MAX+PLUS II Advanced Synthesis software not use timing information from the Fitter to improve the quality of synthesis.

Can't implement option(s) assigned to entity <*name*> due to tri-state connections between hierarchies

---

CAUSE: You assigned an entity-specific logic option to the specified entity. Implementing this entity-specific logic option requires hierarchical synthesis. However, the tri-state connection to the top-level entity prevents hierarchical synthesis. As a result, the MAX+PLUS II Advanced Synthesis software cannot implement the option assignment.

ACTION: Remove the entity-specific logic option or place the tri-state bus entirely within a single entity.

Can't implement Technology Mapper logic option due to netlist optimizations using Fitter timing information

---

CAUSE: You attempted to optimize netlists by directing the MAX+PLUS II Advanced Synthesis software to use timing information from the Fitter to improve the quality of synthesis. However, this netlist optimization is incompatible with the **Technology Mapper** logic option when its value is set to either **Product Term** or **ROM**.

ACTION: If you want to use Fitter timing information for netlist optimization, change the value of the **Technology Mapper** logic option to **Look-up Table (LUT)**. Otherwise, specify that the MAX+PLUS II Advanced Synthesis software not use timing information from the Fitter to improve the quality of synthesis.

Can't implement Technology Mapper logic option due to netlist optimizations using Fitter timing information

---

CAUSE: You attempted to optimize netlists by directing the MAX+PLUS II Advanced Synthesis software to use timing information from the Fitter to improve the quality of synthesis. However, this option is incompatible with the **Technology Mapper** logic option when its value is set to **Auto**.

ACTION: If you want to continue using the **Technology Mapper** logic option when its value set to **Auto**, specify that the MAX+PLUS II Advanced Synthesis software not use timing information from the Fitter to improve the quality of synthesis. Otherwise, the MAX+PLUS II Advanced Synthesis software automatically changes the **Technology Mapper** logic option value to **Look-up Table (LUT)**.

Can't map design hierarchy <*name*> to product term logic because hierarchy contains LUTs

---

CAUSE: You assigned the logic of the specified design hierarchy to product term logic in an Embedded System Block (ESB). However, the Logic Synthesizer cannot map the hierarchy to product term logic because it found LUTs in the design hierarchy. These LUTs are most likely contained in the EDIF Input File (**.edf**) that you are using. The Logic Synthesizer will automatically implement the logic of the specified hierarchy using LUTs.

ACTION: If you want the design hierarchy mapped to LUTs, no action is required. If you want to map the design hierarchy to product term logic, change the design so it does not include LUT logic. Then recompile the design using the non-EDIF design files.

Can't move assignments because of conflicting location assignments

---

CAUSE: The MAX+PLUS II Advanced Synthesis software cannot move an assignment to a location because it conflicts with another assignment. As a result, the name of the logic on the first specified assignment will be lost. To keep the name of the logic on the first specified assignment, the MAX+PLUS II Advanced Synthesis software will move the assignments to another location that shares the same net as the logic that is being removed.

ACTION: Remove or change one of the conflicting assignments.

Can't open database file <*name*>. Database error: <*text*>

CAUSE: The MAX+PLUS II Advanced Synthesis software cannot open the specified database file because you do not have permission to read the file or the directory in which the database file is located.

ACTION: Make sure you have permission to read the database file and the directory in which the file is located and recompile the design.

Can't perform hierarchical synthesis on <*name*> entity due to bidirectional connections between hierarchies

---

CAUSE: You set the **Preserve Hierarchical Boundary** option to Firm or Relaxed for the specified entity. However, the bidirectional connection to the top-level entity requires that the lower-level entity be flattened with the top-level entity. As a result, the Logic Synthesizer cannot perform hierarchical synthesis on the entity.

ACTION: Set the **Preserve Hierarchical Boundary** option to **off**, or place the tri-state bus entirely within a single entity.

Can't perform hierarchical synthesis on <*name*> entity due to netlist optimizations using Fitter timing information

---

CAUSE: You attempted to optimize netlists by directing the MAX+PLUS II Advanced Synthesis software to use timing information from the Fitter to improve the quality of synthesis. However, this option is incompatible with the **Preserve Hierarchical Boundary** logic option assigned to the specified entity because the netlist optimization process flattens the netlist. As a result, the MAX+PLUS II Advanced Synthesis software cannot implement the logic option assignment.

ACTION: If you want to use Fitter timing information for netlist optimization, turn off the **Preserve Hierarchical Boundary** option. Otherwise, specify that the MAX+PLUS II Advanced Synthesis software not use timing information from the Fitter to improve the quality of synthesis.

Can't perform hierarchical synthesis on <*name*> entity due to tri-state connections between hierarchies

---

CAUSE: You set the **Preserve Hierarchical Boundary** option to Firm or Relaxed for the specified entity. However, the bidirectional connection to the top-level entity requires that the lower-level entity be flattened with the top-level entity. As a result, the Logic Synthesizer cannot perform hierarchical synthesis on the entity.

ACTION: Set the **Preserve Hierarchical Boundary** option to **off**, or place the tri-state bus entirely within a single entity.

Can't place carry or cascade chain that starts with logic cell <*name*> due to secondary signal requirements

---

CAUSE: The MAX+PLUS II Advanced Synthesis software cannot implement the carry or cascade chain that starts with the specified logic cell because there are not enough routing resources to route the carry or cascade chain secondary input signals. In FLEX 10KE devices, a carry or cascade chain is placed in LABs such that each LAB uses no more than two clock signals, one asynchronous clear signal, and one signal used as a clear or asynchronous load signal.

ACTION: Modify the design so that the carry or cascade chain can be placed so that each LAB uses no more than two clock signals, one asynchronous clear signal, and one signal used as a clear or asynchronous load signal.

Can't place CARRY primitive <*name*> -- logic cell requires more inputs than it can contain

---

CAUSE: The Compiler cannot place the `CARRY_SUM` primitive because there are not enough LUT inputs in the logic cell. The current design requires more fan-in to the LUT inputs in the logic cell than can be implemented in the logic cell.

ACTION: Check the fan-in to the logic feeding the `CARRY_SUM` primitive and modify the design by removing LUT inputs to the logic cell, or by modifying the fan-in to the logic cell so it can share more signals.

Can't place logic fed by CARRY_SUM primitive <*name*> into a single logic cell

---

CAUSE: The specified `CARRY_SUM` primitive contains fan-out to logic that does not fit in a single logic cell. All fan-out logic in CARRY_SUM primitives must fit on a single logic cell.

ACTION: Modify the design by moving the fan-out to the sum output of the `CARRY_SUM` primitive or simplify the logic fed by the `CARRY_SUM` primitive.

Can't place logic feeding CARRY_SUM primitive <*name*> in single logic cell

---

CAUSE: The specified `CARRY_SUM` primitive contains fan-in from logic that fails to fit in a single logic cell. All fan-in logic in CARRY_SUM primitives must fit on a single logic cell.

ACTION: Modify the design by simplifying the logic that feeds the `CARRY_SUM` primitive to contain fewer inputs to the `CARRY_SUM` primitive.

Can't process state machine <*name*> using user-encoded method

---

CAUSE: You set the **State Machine Processing** logic option to **User-Encoded**; however, the Logic Synthesizer could not find unique user encoding in the design for the specified state machine.

ACTION: No action is required. The Logic Synthesizer automatically chooses the best encoding method for the state machine. If you want to use user-encoded state machine processing, add unique encoding to the design file containing the specified state machine.

Can't read database directory <*name*>

CAUSE: The MAX+PLUS II Advanced Synthesis software cannot read the specified database directory because you do not have permission.

ACTION: Make sure you have permission to read the database file and the directory in which the file is located and recompile the design.

Can't read database file <*name*>

CAUSE: The MAX+PLUS II Advanced Synthesis software cannot read the specified database file because you do not have permission to read the database file or the files directory in which the database file is located.

ACTION: Make sure you have permission to read the database file and the directory in which the file is located and recompile the design.

Can't read or find LMF file <*name*>

CAUSE: You selected the specified Library Mapping File (**.lmf**) for use by the Compiler in compiling EDIF Input Files (**.edf**) that contain non-MAX+PLUS II functions that need to be mapped to MAX+PLUS II functions. However, the MAX+PLUS II Advanced Synthesis software cannot find the specified file. The LMF may be missing, or you may have specified an incorrect file name or path name.

ACTION: Specify the correct file name or path name for the LMF, or copy the LMF to the appropriate directory.

Can't recognize value for <*name*> parameter for WYSIWYG RAM primitive <*name*>

CAUSE: The MAX+PLUS II Advanced Synthesis Compiler cannot recognize the parameter value specified for the specified WYSIWYG RAM primitive.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

Can't remove incompatible database file <*name*> from database directory

---

CAUSE: The MAX+PLUS II Advanced Synthesis software attempted to remove the specified database file because it is incompatible with the current version of the MAX+PLUS II Advanced Synthesis software. However, the MAX+PLUS II Advanced Synthesis software cannot remove the file because you do not have proper read or write permission.

ACTION: Make sure you have read or write permission for the database file.

Can't start <*name*> while the MAX+PLUS II Advanced Synthesis software is running a process or is in batch mode

---

CAUSE:	You attempted to run the specified process, but the MAX+PLUS II Advanced Synthesis software is already running a process, or you used a Tcl script in which a `start_batch` command was called without an `end_batch` command.
ACTION:	If the process is already running, wait for it to end. Check the Tcl script for an unmatched `start_batch` command.

Can't use minimal number of bits to encode state machine *<name>*

CAUSE: You set the **State Machine Processing** logic option to **Minimal Bits**; however, the logic of the specified state machine is too complex for the Logic Synthesizer to process it using the minimal number of bits.

ACTION: No action is required. The Logic Synthesizer processes the state machine using simple binary encoding. To avoid receiving this message in the future, try simplifying the logic of the state machine.

Can't write to database directory <*name*>

CAUSE: The MAX+PLUS II Advanced Synthesis software cannot write to the specified database directory because you do not have permission to write to the specified directory.

ACTION: Change the read or write permissions for directory and recompile the design.

Can't write to database file <*name*>

CAUSE: The MAX+PLUS II Advanced Synthesis software cannot write to the specified database file because you do not have permission to read or write to the specified file or the directory in which the database file is located.

ACTION: Change the read or write permissions for the file and directory and recompile the design.

Can't write to database file <*name*>. Database error: <*text*>

CAUSE: The MAX+PLUS II Advanced Synthesis software cannot write to the specified database file because you do not have permission to write to the specified file or the directory in which the database file is located.

ACTION: Make sure you have permission to write to the database file and the directory in which the file is located and recompile the design.

Can't write to settings and configuration file <*name*>

CAUSE: You attempted to write to the specified settings and configuration file using the `export_settings_files` command-line argument. However, the file is designated as read-only, in use, or some other error occurred while writing to the file.

ACTION: If the file is read-only, you do not have permission to read or write to the file. Change the properties of the file so that it is a writable file and try writing to the file again.

Carry chain that starts with CARRY primitive <*name*> requires <*number*> clear signals, but device can only contain <*number*> clear signals

---

CAUSE: The carry chain that starts with the specified CARRY primitive requires more clear signals than the device can contain. As a result, the MAX+PLUS II Advanced Synthesis software cannot place the carry chain.

ACTION: Modify the design by breaking the chain into smaller chains that require fewer clear signals or by reducing the number of clear signals required by the specified carry chain. You can also change the maximum length of the carry chain with the **Carry Chain Length** option.

Carry chain that starts with CARRY primitive <*name*> too long for the current device

---

CAUSE: The carry chain that starts with the specified CARRY primitive is too long for the current device.

ACTION: Modify the design by breaking the chain into smaller chains or change the maximum length of the carry chain with the **Carry Chain Length** option..

CARRY primitive <*name*> is part of cyclic carry chain

CAUSE: In the current design, the specified CARRY primitive is part of a cyclic carry chain. However, the Compiler cannot implement a cyclic carry chain.

ACTION: Modify the design so the carry chain is non-cyclic.

CASCADE primitive <*name*> contains fan-out to more than one destination

---

CAUSE: The specified `CASCADE` primitive contains fan-out to multiple destinations. However, `CASCADE` primitives should contain fan-out to a single destination.

ACTION: Modify the design so the fan-out is on the input side of the `CASCADE` primitive, so the primitive contains fan-out to a single destination.

CASCADE primitive <*name*> fed by VCC or GND

CAUSE: The specified `CASCADE` primitive is fed by `VCC` or `GND`, or logic that reduces to `VCC` or `GND`.

ACTION: Modify the logic in the design so the `CASCADE` primitive is not fed by `VCC` or `GND`, or remove the `CASCADE` primitive from the design to improve minimization.

CASCADE primitive <*name*> is fed by non-combinatorial logic

CAUSE: The specified CASCADE primitive is fed by non-combinatorial logic. However, a CASCADE primitive can be fed by only combinatorial logic.

ACTION: Modify the design so combinatorial logic feeds the CASCADE primitive, or remove the CASCADE primitive from the design.

CASCADE primitive <*name*> is part of cyclic cascade chain

CAUSE: In the current design, the specified CASCADE primitive is part of a cyclic cascade chain. However, the Compiler cannot implement a cyclic cascade chain.

ACTION: Modify the design so the cascade chain is non-cyclic.

Cascin port of WYSIWYG LCELL primitive <*name*> is connected to gate primitive <*name*>, but must be connected to a WYSIWYG LCELL

CAUSE: The `cascin` port of the specified WYSIWYG LCELL primitive is connected to the specified gate primitive, but a WYSIWYG LCELL can only cascade in from another WYSIWYG LCELL primitive.

ACTION: Change the cascade out port to a normal input or replace the source gate with a WYSIWYG LCELL primitive.

Cascout port of WYSIWYG LCELL primitive <*name*> is connected to gate primitive <*name*>, but must be connected to a WYSIWYG LCELL

CAUSE: The `cascout` port of the specified WYSIWYG LCELL primitive is connected to the specified gate primitive, but a WYSIWYG LCELL can only cascade out to another WYSIWYG LCELL primitive.

ACTION: Change the `cascout` port to a normal output or replace the destination gate primitive with a WYSIWYG LCELL primitive.

Changed operation mode of WYSIWYG I/O primitive *<name>* from type *<type>* to type *<type>* to match type of pin *<name>* connected to its padio port

---

CAUSE: The specified WYSIWYG I/O primitive operation mode does not match the type of the pin connected to its `padio` port. The MAX+PLUS II Advanced Synthesis software converted the operation mode from the specified pin type to the specified matching pin type.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

Changed operation mode of WYSIWYG I/O primitives to match their padio pin type

---

CAUSE: The specified WYSIWYG I/O primitive operation modes do not match the type of the pins that connect to their `padio` ports, but the MAX+PLUS II Advanced Synthesis software converted the operation modes to this pin type.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

Cin port of WYSIWYG LCELL primitive <*name*> connected to gate primitive <*name*>, but must be connected to a WYSIWYG LCELL

CAUSE: The `cin` port of the specified WYSIWYG LCELL primitive is connected to the specified gate primitive, but a WYSIWYG LCELL primitive can only carry in from another WYSIWYG LCELL primitive.

ACTION: Change the `cin` port to a normal input or replace the source gate primitive with a WYSIWYG LCELL primitive.

clk0 port of WYSIWYG primitive <*name*> cannot be specified if it is not used

---

CAUSE: The specified WYSIWYG primitive has a `clk0` port, but the port is not in use. The WYSIWYG primitive cannot have a `clk0` port if the port is not in use. The `CLOCK0` parameter value must be either specified for the WYSIWYG primitive or removed from the primitive.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

clk1 port of WYSIWYG primitive <*name*> cannot be specified if it is not used

---

CAUSE: The specified WYSIWYG primitive has a `clk1` port, but the port is not in use. The WYSIWYG primitive cannot have a `clk1` port if the port is not in use. The `CLOCK1` parameter value must be either specified for the WYSIWYG primitive or removed from the primitive.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

CLKLOCK primitive <*name*> has illegal ClockBoost parameter value of <*number*>, but must be a value of 1 or 2

CAUSE: The design contains the specified `CLKLOCK` primitives that has a ClockBoost parameter set to a value other than 1 or 2. However, the ClockBoost parameter can have only a value of 1 or 2.

ACTION: Change the ClockBoost parameter to a legal value of 1 or 2.

CLKLOCKx1 Input Frequency logic option can turned on for only input pins, but was turned on for non-input pin *<name>*

CAUSE: You turned on the VCC**CLKLOCKx1 Input Frequency** option for the specified input pin.However, you can use the **CLKLOCKx1 Input Frequency** option only for input pins.

ACTION: Turn off the logic option for the node.

ClockLock PLL <*name*> has an illegal CLK0_DIVIDE_BY or CLK1_DIVIDE_BY parameter value <*number*> for <*name*> output port

---

CAUSE: You specified an illegal value for the specified ClockLock PLL's `CLK0_DIVIDE_BY` or `CLK1_DIVIDE_BY` parameter for the specified output port. The `CLK0_DIVIDE_BY` or `CLK1_DIVIDE_BY` parameter value for a given port must be an integer greater than zero. This error usually occurs when you instantiate a megafunction directly in a text file rather than using the **MegaWizard Plug-In Manager** (Tools menu).

ACTION: Locate the source of the message to determine which megafunction parameter is causing the error. To avoid receiving this message in the future, Altera recommends using the **MegaWizard Plug-In Manager**. Specify a legal parameter value.

ClockLock PLL <*name*> has illegal MULTIPLY_BY parameter value <*number*> for <*name*> output port

---

CAUSE: You specified an illegal value for the specified ClockLock PLL's `CLK0_MULTIPLY_BY` or `CLK1_MULTIPLY_BY` parameter for the specified output port. The `CLK0_MULTIPLY_BY` or `CLK1_MULTIPLY_BY` parameter value for a given port must be an integer greater than zero. This error usually occurs when you instantiate a megafunction directly in a text file rather than using the **MegaWizard Plug-In Manager** (Tools menu).

ACTION: Locate the source of the message to determine which megafunction parameter is causing the error. To avoid receiving this message in the future, Altera recommends using the **MegaWizard Plug-In Manager**. Specify a legal parameter value.

ClockLock PLL *<name>* has illegal or non-numeric value *<text>* for INPUT_FREQUENCY parameter

---

CAUSE: The specified ClockLock PLL's `INPUT_FREQUENCY` parameter has an illegal or non-numeric value. The `INPUT_FREQUENCY` parameter value must be a legal number. This error usually occurs when you instantiate a megafunction directly in a text file rather than using the **MegaWizard Plug-In Manager** (Tools menu).

ACTION: Locate the source of the message to determine which megafunction parameter is causing the error and specify a legal numeric parameter value. You can also add a time unit or frequency unit to the number. If you do not specify a time unit or frequency unit, the Compiler uses the ps (picoseconds) time unit. To avoid receiving this message in the future, Altera recommends using the **MegaWizard Plug-In Manager** to specify legal parameter values for megafunctions.

ClockLock PLL <*name*> has illegal value <*number*> for INVALID_LOCK_MULTIPLIER parameter

---

CAUSE: You specified an illegal value for the ClockLock PLL's `INVALID_LOCK_MULTIPLIER` parameter value. The legal values for the `INVALID_LOCK_MULTIPLIER` parameter are 1 and 5. This error usually occurs when you instantiate a megafunction directly in a text file rather than using the **MegaWizard Plug-In Manager** (Tools menu).

ACTION: Locate the source of the message to determine which megafunction parameter is causing the error. To avoid receiving this message in the future, Altera recommends using the **MegaWizard Plug-In Manager**. Specify a legal parameter value.

ClockLock PLL <*name*> has illegal value <*number*> for LOCK_HIGH parameter

---

CAUSE: You specified an illegal value for the ClockLock PLL's `LOCK_HIGH` parameter. The legal values for the `LOCK_HIGH` parameter must be integers greater than 0. This error usually occurs when you instantiate a megafunction directly in a text file rather than using the **MegaWizard Plug-In Manager** (Tools menu).

ACTION: Locate the source of the message to determine which megafunction parameter is causing the error. To avoid receiving this message in the future, Altera recommends using the **MegaWizard Plug-In Manager**. Specify a legal parameter value.

ClockLock PLL <*name*> has illegal value <*number*> for LOCK_LOW parameter

---

CAUSE: You specified an illegal value for the ClockLock PLL's `LOCK_LOW` parameter. The legal value must be an integer greater than zero. This error usually occurs when you instantiate a megafunction directly in a text file rather than using the **MegaWizard Plug-In Manager** (Tools menu).

ACTION: Locate the source of the message to determine which megafunction parameter is causing the error. To avoid receiving this message in the future, Altera recommends using the **MegaWizard Plug-In Manager**. Specify a legal parameter value.

ClockLock PLL <*name*> has illegal value <*number*> for VALID_LOCK_MULTIPLIER parameter

---

CAUSE: You specified an illegal value for the ClockLock PLL's `VALID_LOCK_MULTIPLIER` parameter. The legal values for the `VALID_LOCK_MULTIPLIER` parameter are 1 and 5. This error usually occurs when you instantiate a megafunction directly in a text file rather than using the **MegaWizard Plug-In Manager** (Tools menu).

ACTION: Locate the source of the message to determine which megafunction parameter is causing the error. To avoid receiving this message in the future, Altera recommends using the **MegaWizard Plug-In Manager**. Specify a legal parameter value.

ClockLock PLL <*name*> has illegal value <*text*> for PHASE_SHIFT parameter

---

CAUSE: he specified ClockLock PLL's `PHASE_SHIFT` parameter is set to an illegal value. The `PHASE_SHIFT` parameter value must be a legal number. This error usually occurs when you instantiate a megafunction directly in a text file rather than using the **MegaWizard Plug-In Manager** (Tools menu).

ACTION: Locate the source of the message to determine which megafunction parameter is causing the error and specify a legal numeric parameter value. You can also add a time unit or frequency unit to the number. If you do not specify a time unit or frequency unit, the Compiler uses the ps (picoseconds) time unit. To avoid receiving this message in the future, Altera recommends using the **MegaWizard Plug-In Manager**. Specify a legal parameter value.

ClockLock PLL <*name*> must have value for CLOCK1_BOOST or CLOCK2_BOOST parameter

---

CAUSE: You did not specify a value for the specified ClockLock PLL's `CLOCK1_BOOST` or `CLOCK2_BOOST` parameter. A `CLOCK1_BOOST` or `CLOCK2_BOOST` parameter value must be specified. This error usually occurs when you instantiate a megafunction directly in a text file rather than using the **MegaWizard Plug-In Manager** (Tools menu).

ACTION: Locate the source of the message to locate the megafunction parameter that is causing the error, and specify a legal parameter value. To avoid receiving this message in the future, Altera recommends using the **MegaWizard Plug-In Manager** to specify legal parameter values for megafunctions.

ClockLock PLL <*name*> must have value for CLOCK1_BOOST or CLOCK2_BOOST parameter

---

CAUSE: You did not specify a value for the specified ClockLock PLL's `CLOCK1_BOOST` or `CLOCK2_BOOST` parameter. A `CLOCK1_BOOST` or `CLOCK2_BOOST` parameter value must be specified. This error usually occurs when you instantiate a megafunction directly in a text file rather than using the **MegaWizard Plug-In Manager**.

ACTION: Locate the source of the message to locate the megafunction parameter that is causing the error, and specify a legal parameter value. To avoid receiving this message in the future, Altera recommends using the **MegaWizard Plug-In Manager** to specify legal parameter values for megafunctions.

ClockLock PLL <*name*> must have value for INPUT_FREQUENCY parameter

---

CAUSE: You did not specify a value for the specified ClockLock PLL's `INPUT_FREQUENCY` parameter. An `INPUT_FREQUENCY` parameter value must be specified. This error usually occurs when you instantiate a megafunction directly in a text file rather than using the **MegaWizard Plug-In Manager**.

ACTION: Locate the source of the message to determine which megafunction parameter is causing the error. To avoid receiving this message in the future, Altera recommends using the **MegaWizard Plug-In Manager**. Specify a legal parameter value, or edit input clock settings to specify the frequency.

ClockLock PLL <*name*> must have value for INPUT_FREQUENCY parameter

---

CAUSE: You did not specify a value for the specified ClockLock PLL's `INPUT_FREQUENCY` parameter. An `INPUT_FREQUENCY` parameter value must be specified. This error usually occurs when you instantiate a megafunction directly in a text file rather than using the **MegaWizard Plug-In Manager** (Tools menu).

ACTION: Locate the source of the message to locate the megafunction parameter that is causing the error, and specify a legal parameter value. To avoid receiving this message in the future, Altera recommends using the **MegaWizard Plug-In Manager** to specify legal parameter values for megafunctions.

ClockLock PLL <*name*> must have value for INPUT_FREQUENCY parameter

---

CAUSE: You did not specify a value for the specified ClockLock PLL's `INPUT_FREQUENCY` parameter. An `INPUT_FREQUENCY` parameter value must be specified. This error usually occurs when you instantiate a megafunction directly in a text file rather than using the **MegaWizard Plug-In Manager**.

ACTION: Locate the source of the message to locate the megafunction parameter that is causing the error, and specify a legal parameter value. To avoid receiving this message in the future, Altera recommends using the **MegaWizard Plug-In Manager** to specify legal parameter values for megafunctions.

ClockLock PLL cannot have unsupported value for OPERATION_MODE parameter

---

CAUSE: You specified an unsupported value for an `OPERATION_MODE` parameter in a ClockLock PLL. Supported parameter values are `NORMAL`, `ZERO_DELAY_BUFFER`, `EXTERNAL_FEEDBACK`, and `LVDS`. This error usually occurs when you instantiate a megafunction directly in a text file rather than using the **MegaWizard Plug-In Manager** command (Tools menu).

ACTION: Locate the source of the message to determine which megafunction parameter is causing the error. To avoid receiving this message in the future, Altera recommends using the **MegaWizard Plug-In Manager** command. Specify a legal value for the parameter.

ClockLock PLL must contain input clock signal

CAUSE: The ClockLock PLL in the design file does not contain a clock signal because its `clkin` port is not connected.

ACTION: Connect the `clkin` port in the ClockLock PLL in the design file.

ClockLock PLL or WYSIWYG primitive <*name*> has illegal time delay parameter value <*number*> for output <*name*> port

---

CAUSE 1: You specified an illegal value for one of the specified ClockLock PLL's time delay parameters (that is, `CLOCK0_TIME_DELAY`, `CLOCK1_TIME_DELAY`, `CLOCK2_TIME_DELAY`, or `CLOCK_EXT_TIME_DELAY` parameters) for the megafunction's specified output port. The time delay value must be between -2 ns and +2 ns, with a resolution of 0.25 ns. This error usually occurs when you instantiate a megafunction directly in a text file rather than using the **MegaWizard Plug-In Manager** (Tools menu).

ACTION: Locate the source of the message to determine which megafunction parameter is causing the error. To avoid receiving this message in the future, Altera recommends using the **MegaWizard Plug-In Manager**. Specify a legal parameter value.

CAUSE 2: The specified WYSIWYG primitive has an illegal value for a time delay parameter (that is, `CLK0_TIME_DELAY`, `CLK1_TIME_DELAY`, `CLK2_TIME_DELAY`, or `EXTCLK_TIME_DELAY` parameters) for the WYSIWYG primitive's specified output port. The time delay value must be between -2 ns and +2 ns, with a resolution of 0.25 ns. The illegal parameter value must be replaced with a legal parameter value.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

ClockLock PLL parameter value <*text*> is not in range allowed for parameter

---

CAUSE: You specified a value for an ClockLock PLL parameter, but the value is not in the range allowed for the value. This error usually occurs when you instantiate a megafunction directly in a text file, instead of using the **MegaWizard Plug-In Manager** (Tools menu).

ACTION: Locate the source of the message to determine which megafunction parameter is causing the error. To avoid receiving this message in the future, Altera recommends using the **MegaWizard Plug-In Manager**. Specify a legal parameter value.

ClockLock PLL's PHASE_SHIFT parameter value cannot be set to non-0 phase shift when OPERATION_MODE parameter is not set to NORMAL

CAUSE: You specified a non-0 phase shift value for the `PHASE_SHIFT` parameter in the ClockLock PLL when its `OPERATION_MODE` parameter is not set to `NORMAL`. You can only specify a non-0 phase shift value for the `PHASE_SHIFT` parameter when the ClockLock PLL's `OPERATION_MODE` parameter is set to `NORMAL`. This error usually occurs when you instantiate a megafunction directly in a text file rather than using the **MegaWizard Plug-In Manager** (Tools menu).

ACTION: Locate the source of the message to determine which megafunction parameter is causing the error and specify a legal phase shift value for the ClockLock PLL. To avoid receiving this message in the future, Altera recommends using the **MegaWizard Plug-In Manager** to specify legal parameter values for megafunctions.

ClockLock PLL's PHASE_SHIFT parameter value set to inconsistent phase shifts between clk0 and clk1 ports

---

CAUSE: You specified inconsistent phase shifts for a ClockLock PLL's `PHASE_SHIFT` parameter between the `clk0` and `clk1` ports.

ACTION: Specify consistent phase shifts for the ClockLock PLL.

clr0 port of WYSIWYG primitive <*name*> cannot be specified if it is not used

---

CAUSE: The specified WYSIWYG primitive has a `clr0` port, but the port is not in use. The WYSIWYG primitive cannot have a `clr0` port if the port is not in use. The `CLEAR0` parameter value must be either specified for the WYSIWYG primitive or removed from the primitive.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

clr1 port of WYSIWYG primitive <*name*> cannot be specified if it is not used

---

CAUSE: The specified WYSIWYG primitive has a `clr1` port, but the port is not connected. The WYSIWYG primitive cannot have a `clr1` port if the port is not connected. The `CLEAR1` parameter value must be either specified for the WYSIWYG primitive or removed from the primitive.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

Compilation or simulation focus name <*name*> specified for <*Compiler or Simulator*> settings <*name*> contains illegal characters

---

CAUSE: You selected the specified design entity as the compilation or simulation focus for the specified Compiler or Simulator settings, but the name of the entity contains illegal name characters. The MAX+PLUS II Advanced Synthesis software ignored the compilation or simulation focus.

ACTION: Change the name of the design entity or specify a different design entity as the compilation focus or simulation focus.

Compiler inserted the following logic cells to protect hierarchy boundaries

---

CAUSE: You assigned the **Preserve Hierarchical Boundary** logic option to one or more subdesigns in the current project. To honor this assignment, the Compiler inserted logic cells which behave as simple wires. You can delete the hierarchical assignments to remove these logic cells to improve fitting and performance.

ACTION: No action is required. Refer to the submessage(s) in the **Messages** window to determine which cells are added.

Completed encoding using <*number*> state bits

CAUSE: The Logic Synthesizer completed encoding a state machine and is reporting that it used the specified total number of state bits to encode the state machine.

ACTION: No action is required. Click the **+** icon next to this message in the Messages window to display the encoding details.

Converted <*number*> single input CARRY primitives to CARRY_SUM primitives

---

CAUSE: The Compiler replaced the specified number of single input `CARRY` primitives with the newer `CARRY_SUM` primitive. The `CARRY` primitive is provided only for backward compatibility with MAX+PLUS II designs. New projects should use `CARRY_SUM` primitives exclusively.

ACTION: No action is required.

Converted presettable and clearable register to equivalent circuits with latches. Registers will power up to an undefined state, and DEVCLRn will place the registers in an undefined state.

CAUSE: A register in a project targeted for one or more APEX 20K, APEX II, or ARM-based Excalibur, FLEX 6000 device(s) has one of the following conditions:

- The register has both a preset and a clear signal.
- The register has a preset signal but does not have a clear signal, and the MAX+PLUS II Advanced Synthesis Compiler turned off the `NOT` Gate Push-Back logic option.

ACTION: No action is required. As a result, the MAX+PLUS II Advanced Synthesis Compiler converted the register to equivalent circuits with latches. However, the resulting register will power up to an undefined state (X). Also, DEV_CLRn will place the register in an undefined state.

Converted TRI buffer to OR gate or removed OPNDRN

CAUSE: The Compiler converted all `TRI`s to `OR` gates and removed all `OPNDRN`s that feed logic gates.

ACTION: No action is required.

Converting TRI node <*name*> that feeds logic to an OR gate

CAUSE: The Compiler is converting `TRI(IN, OE)` to `IN+!OE`.

ACTION: No action is required.

Core <*name*> is not enabled for current device family

CAUSE: You instantiated the specified core and attempted to compile the design. However, the specified core cannot be used with the selected device family. As a result, the MAX+PLUS II Advanced Synthesis software stopped the compilation.

ACTION: Refer to the core documentation for information about which device families can be used with the specified core. If necessary, change the device family and recompile the design. A later version of this core may support this device family; visit the IP MegaStore in the Altera web site or in the MegaWizard Portal extension to the **MegaWizard Plug-in Manager** (Tools menu) to check for later versions.

Cout port of WYSIWYG LCELL primitive <*name*> is connected to gate primitive <*name*>, but must be connected to a WYSIWYG LCELL

CAUSE: The `cout` port of the specified WYSIWYG LCELL primitive is connected to the specified gate primitive, but a WYSIWYG LCELL primitive can only carry out to other WYSIWYG LCELL primitives.

ACTION: Change the `cout` port to a normal output or replace the destination gate primitive with a WYSIWYG LCELL primitive.

Created node <*name*> as a RAM by implementing register logic with Embedded System Block. Functionality differs from the original design.

CAUSE: In a Verilog Design File (**.v**) or VHDL Design File (**.vhd**), you specified a set of registers that act as a RAM, and specified that data is written to the RAM on the positive edge of the RAM's write clock. When the Compiler created the specified node as a RAM using the registers, it converted the registers into a RAM megafunction to implement the register logic with an Embedded System Block (ESB) on the target device you specified for the current design. However, data is written to the ESB on the negative edge of the ESB's write clock. Because the original design specified that data is written to the RAM on the positive edge of the RAM's write clock, the functionality of the ESB is not identical to the functionality of the RAM in the original design.

ACTION: If you intended the Compiler to convert the registers into a RAM megafunction to implement the register logic with an ESB, and you do not mind if the functionality of the ESB is not identical to the functionality of the RAM in the original design, no action is required. If you want the Compiler to use a RAM megafunction, but you want to avoid receiving this message in the future, replace the registers and address logic in the Verilog Design File or VHDL Design File with an explicit instantiation of a RAM megafunction. If you want the functionality of the RAM in the original design to be the same as functionality that is implemented in the target device, prevent the Compiler from converting the registers into a RAM megafunction by turning off the **Auto RAM Replacement** logic option.

Database file <*name*> corrupted. Database error: <*text*>

CAUSE: The MAX+PLUS II Advanced Synthesis software cannot read the specified database file due to the specified error. This message can occur if the specified file was manually edited or renamed, another user tried to access the file, or the current version of the MAX+PLUS II Advanced Synthesis software is not compatible with the specified file.

ACTION: Delete the specified database file and recompile the design Deleting the database file erases the results of the previous compilation; recompiling the design restores the database file.

Database file <*name*>, created by MAX+PLUS II Advanced Synthesis <*version number*> software, not compatible with current MAX+PLUS II Advanced Synthesis <*version number*> software

---

CAUSE: The specified database file is incompatible with the specified current version of the MAX+PLUS II Advanced Synthesis software because it was created by the specified previous version of the MAX+PLUS II Advanced Synthesis software. This message can occur if you recently upgraded the MAX+PLUS II Advanced Synthesis software.

ACTION: Delete the specified database file and recompile the design. Deleting the database file erases the results of the previous compilation; recompiling the design restores the database file.

Design contains <*number*> input pin(s) that do not drive logic

---

CAUSE: The specified number of input pins in the design do not drive any logic. However, all input pins in the design should drive logic.

ACTION: If this condition is intentional, no action is required. If this condition is unintentional, make sure all input pins in the design drive logic.

Design contains <*number*> virtual pins

CAUSE: You assigned the **Virtual Pin** logic option to the specified number of nodes in the design.

ACTION: No action is required.

DEV_CLRn pin will set, and not reset, register with preset signal due to NOT Gate Push-Back

---

CAUSE: You enabled the device-wide reset (DEV_CLRn) pin for the current design's target device. As a result, the MAX+PLUS II Advanced Synthesis Compiler enabled the NOT Gate Push-Back on some registers in the current project. For devices that have the DEV_CLRn option, these registers will be set rather than reset when DEV_CLRn is enabled.

ACTION: No action is required. However, if you want the MAX+PLUS II Advanced Synthesis software to be reset rather than set the registers with preset signals, turn off the NOT Gate Push-Back option for those registers.

Disk is full -- current compilation halted

CAUSE: You attempted to compile the current design; however, the MAX+PLUS II Advanced Synthesis software cannot complete the compilation because the disk containing the database files is full.

ACTION: Create disk space on the drive that contains the project or save the project to a different drive and then recompile the design.

Disk space full error at Logic Synthesizer stage <*text*>: <*text*>

CAUSE: You attempted to compile a design. However, you do not have enough disk space to complete the compilation at the specified Compiler module.

ACTION: Create disk space on your hard drive and recompile the design.

Duplicate LATCH primitives merged into single LATCH primitive

---

CAUSE: In a design file in the current design, you created duplicate `LATCH` primitives and compiled the design. As a result, the Compiler merged the duplicate `LATCH` primitives into a single `LATCH` primitve.

ACTION: No action is required.

Duplicate LATCH primitives merged into single LATCH primitive <*name*>

CAUSE: In a design file in the current design, you created duplicate `LATCH` primitives and compiled the design. As a result, the Compiler merged the duplicate `LATCH` primitives into the specified `LATCH` primitve.

ACTION: No action is required.

Duplicate registers merged to single register

CAUSE: You created duplicate registers in the network, so the Compiler merged them into a single register.

ACTION: No action is required. To prevent duplicate registers from merging into a single register, turn off the **Remove Duplicate Registers** logic option for this register.

Duplicate registers merged to single register <*name*>

CAUSE: You created duplicate registers, so the Compiler merged them into a single register.

ACTION: No action is required. To prevent duplicate registers from merging into a single register, turn off the **Remove Duplicate Registers** logic option on this register.

Duplicate registers merged to single register <*name*>, power-up level has changed

---

CAUSE: You created duplicate registers, so the Compiler merged them into a single register. The compiler was able to do this because theeir registers were specified as power-up don't care.

ACTION: No action is required. To prevent duplicate registers from merging into a single register, turn off the **Remove Duplicate Registers** logic option on this register. To prevent power-up levels to change, turn off the project wide power-up don't care option, or set the power-up high logic option on this register.

Encoded state bit <*name*>

CAUSE: The Logic Synthesizer is reporting that it encoded a state machine using the specified state bit.

ACTION: No action is required.

Encoding result for state machine <*name*>

CAUSE: You compiled a design that contains the specified state machine. The Logic Synthesizer is reporting the encoding result of the specified state machine.

ACTION: No action is required. Click the **+** icon next to this message in the Messages window to display the encoding details.

Entity <*name*> is instantiated by entity <*name*>

CAUSE: You attempted to compile the current design. However, the MAX+PLUS II Advanced Synthesis software detected a loop in netlist extraction. The specified entity instantiates itself in the second specified entity. As a result, the netlist extraction was unsuccessful.

ACTION: Modify the design so that the entity does not instantiate itself and recompile the project.

Entity name <*name*> conflicts with MAX+PLUS II Advanced Synthesis primitive name

---

CAUSE: You named the specified entity with the same name as a MAX+PLUS II primitive name. However, the MAX+PLUS II Advanced Synthesis software does not permit you to give the same name to an entity as to a MAX+PLUS II primitive.

ACTION: Modify the name of the entity so it does not conflict with the MAX+PLUS II primitive name, or remove the entity.

Fan-in of WYSIWYG ClockLock PLL primitive <*name*> can fan-out only to WYSIWYG ClockLock PLL primitives with the same input frequency value

---

CAUSE: The input that feeds the specified WYSIWYG ClockLock PLL primitive also fans out to other locations that do not have the same input frequency value as the WYSIWYG Clocklock PLL primitive. However, the fan-in of a WYSIWYG ClockLock PLL primitive can fan-out only to other WYSIWYG ClockLock PLL primitives with the same input frequency value.

ACTION: Edit the design to remove any fan-in to a WYSIWYG ClockLock PLL primitive that does not fan-out to a WYSIWYG ClockLock PLL primitive with the same input frequency value.

Fan-in of WYSIWYG ClockLock PLL primitive <*name*> cannot feed more than two destinations

---

CAUSE: The fan-in of the specified WYSIWYG ClockLock PLL primitive feeds more than two destinations (that is, more than two ClockLock PLL buffers). However, the fan-in of a WYSIWYG ClockLock PLL primitive can feed no more than two destinations.

ACTION: Correct the WYSIWYG ClockLock PLL primitive in the design so the fan-in feeds no more than two destinations.

Feature <*name*> is not available with your current license

CAUSE: You selected a feature in the MAX+PLUS II Advanced Synthesis software, but the software's license file does not support this feature, or the license file does not exist.

ACTION: Click **OK** to close the message dialog box. If you do not want to use the feature, no action is required. Otherwise, you can obtain a MAX+PLUS II Advanced Synthesis software license file from the Licensing section of the Altera web site, which is available at **www.altera.com/licensing.html**.

File <*name*> batch mode was left open from the last script execution. Add an end_batch function to the Tcl script.

CAUSE: You ran a Tcl script using a **start_batch** function, but did not include an **end_batch** function.

ACTION: Add an **end_batch** function to the Tcl script.

File <*name*> is not a recognized design file type

CAUSE: You performed one of the following actions:

- Analyzed a file
- Created an AHDL Include File (**.inc**)
- Created a Block Symbol File (**.bsf**)

However, the MAX+PLUS II Advanced Synthesis software cannot continue the process because the specified file that is required for the process is not a recognized file type.

ACTION: Make sure that the file has the correct file extension and restart the process.

Found <*number*> design units and <*number*> entities in source file <*name*>

CAUSE: You instantiated the specified number of design units in the design, and the Compiler has identified the specified number of entities in the specified source file.

ACTION: No action is required.

Found design unit <*name*>: <*name*>

CAUSE: You instantiated the specified design unit in the design, and the Compiler detected the specified design unit.

ACTION: No action is required.

Found entity <*name*>: <*name*>

CAUSE: You instantiated the specified design unit in the design, and the Compiler detected the specified entity.

ACTION: No action is required.

Found the following redundant logic cells in design

CAUSE: The Compiler found nodes in the current design that are functional duplicates of other nodes. For example, the design may contain duplicate registers that perform the same function. Removing them does not affect the function of the design, and improves area and may improve delay.

ACTION: Refer to the submessage(s) in the **Messages** window to determine which cells are redundant logic cells. To avoid receiving this message in the future, turn on the **Remove Redundant Logic Cells** option

Found the following redundant logic cells in design

CAUSE: The current design contains redundant user cells that act as simple wires. You can remove the redundant logic cells to improve fitting and performance.

ACTION: Refer to the submessage(s) in the **Messages** window to determine which cells are redundant logic cells. To avoid receiving this message in the future, turn on the **Remove Redundant Logic Cells** option.

HSDI Primitive <*name*> not supported

CAUSE: You attempted to use the specified HSDI primitive, but the MAX+PLUS II Advanced Synthesis software does not support the primitive.

ACTION: Contact Altera Applications for information on obtaining the HDSI primitive license.

Ignore assignments for <*name*>[<*number*>] because it is an invalid assignment target -- <*name*> is a single bit, and cannot have members

---

CAUSE: You assigned the specified signal as a target, but the specified bit is a single bit, and cannot have members. As a result, the MAX+PLUS II Advanced Synthesis software is ignoring the assignment.

ACTION: Modify the assignment so that it contains a valid target.

Ignored <*name*> CARRY buffer(s)

CAUSE: The Compiler ignored the specified buffer(s), as specified by the related logic option, or because they are not appropriate for the setting for the selected **technology mapper** logic option.

ACTION: No action is required.

Ignored <*name*> logic option assignment--illegal setting <*value*>

CAUSE: The specified logic option assignment, which appears in a Text Design File (**.tdf**), has a specified illegal setting. The Compiler ignored the assignment.

ACTION: No action is required. To avoid receiving this message in the future, use a legal setting of `On` or `Off` for the logic option.

Ignored <*number*> buffer(s)

CAUSE: The Compiler ignored the specified buffer(s), as requested by the appropriate logic option, or because they are not appropriate for the selected **technology mapper** logic option.

ACTION: No action is required.

Ignored <*number*> CARRY_SUM buffer(s)

CAUSE: The Compiler ignored the specified buffer(s), as requested by the appropriate logic option, or because they are not appropriate for the selected **technology mapper** logic option.

ACTION: No action is required.

Ignored <*number*> CARRY_SUM primitive(s) -- cannot place fan-out logic in single logic cell

---

CAUSE: The MAX+PLUS II Advanced Synthesis software ignored the specified number of `CARRY_SUM` primitives because they contain fan-out to logic that does not fit in a single logic cell. All fan-in logic in CARRY_SUM primitives must fit on a single logic cell.

ACTION: Modify the design by moving the fan-out to the sum output of the `CARRY_SUM` primitive(s) or simplify the logic that the `CARRY_SUM` primitive(s) feed.

Ignored <*number*> CARRY_SUM primitives

CAUSE: The MAX+PLUS II Advanced Synthesis software ignored the specified number of `CARRY_SUM` primitives because they could not be implemented in the target device family you specified for the current design.

ACTION: Refer to warning message(s) in the Messages window to determine the specific reasons the MAX+PLUS II Advanced Synthesis software cannot implement the `CARRY_SUM` primitive(s).

Ignored <*number*> CARRY_SUM primitives -- cannot place fan-in logic in single logic cell

---

CAUSE: The MAX+PLUS II Advanced Synthesis software ignored the specified number of `CARRY_SUM` primitives because they contain fan-in from logic that does not fit in a single logic cell. All fan-in logic in CARRY_SUM primitives must fit on a single logic cell.

ACTION: Modify the design by simplifying the logic that feeds the `CARRY_SUM` primitive to contain fewer inputs.

Ignored <*number*> CARRY_SUM primitives -- logic cell requires more inputs than it can contain

---

CAUSE: The MAX+PLUS II Advanced Synthesis software ignored the specified number of `CARRY_SUM` primitives because there are not enough LUT inputs in the logic cell. The current design requires more fan-in to the LUT inputs in the logic cell than can be implemented in the logic cell.

ACTION: Check the fan-in to the logic feeding the `CARRY_SUM` primitive and modify the design by removing LUT inputs to the logic cell, or by modifying the fan-in to the logic cell so it can share more signals.

Ignored <*number*> CASCADE buffer(s)

CAUSE: The Compiler ignored the specified buffer(s), as requested by the appropriate logic option, or because they are not appropriate for the selected technology mapper.

ACTION: No action is required.

Ignored <*number*> EXPANDER buffer(s)

CAUSE: The Compiler ignored the specified buffer(s), as requested by the appropriate logic option, or because they are not appropriate for the selected **technology mapper** logic option.

ACTION: No action is required.

Ignored <*number*> GLOBAL buffer(s)

CAUSE: The Compiler ignored the specified buffer(s), as requested by the appropriate logic option, or because they are not appropriate for the selected **technology mapper** logic option.

ACTION: No action is required.

Ignored <*number*> LCELL buffer(s)

CAUSE: The Compiler ignored the specified buffer(s), as requested by the appropriate logic option, or because they are not appropriate for the selected **technology mapper** logic option.

ACTION: No action is required.

Ignored <*number*> ROW_GLOBAL buffer(s)

CAUSE: The Compiler ignored the specified buffer(s), as requested by the appropriate logic option, or because they are not appropriate for the selected **technology mapper** logic option.

ACTION: No action is required.

Ignored <*number*> SOFT buffer(s)

CAUSE: The Compiler ignored the specified buffer(s), as requested by the appropriate logic option, or because they are not appropriate for the selected **technology mapper** logic option.

ACTION: No action is required.

Ignored duplicate design unit <*name*> in file <*name*>

CAUSE: You defined multiple design units with the same name, so the MAX+PLUS II Advanced Synthesis software ignored the duplicate design unit.

ACTION: Rename the duplicate design unit or remove the file that defines the duplicate from the project.

Ignored duplicate entity <*name*> found in file <*name*>

CAUSE: You defined multiple design units with the same name, so the MAX+PLUS II Advanced Synthesis software ignored the duplicate design unit.

ACTION: Rename the duplicate entity or remove the file that defines the duplicate from the project.

Ignored duplicate of assignment <*name*> for node <*name*>

CAUSE: The specified assignment appears in a Compiler Settings File (**.csf**) or Entity Settings File (**.esf**) more than once for the specified node. The MAX+PLUS II Advanced Synthesis software ignored and removed the duplicate assignment.

ACTION: No action is required.

Ignored Logic Cell Insertion logic option on <*name*> -- Logic Cell Insertion can only be assigned to non-global direct data path connections

---

CAUSE: You turned on the **Logic Cell Insertion** logic option to add logic cell buffers to the specified node(s) in the design. However, you illegally assigned the logic option to a node other than a non-global direct data path connection. The Logic Cell Insertion logic options can be assigned only to non-global direct data path connections.

ACTION: Use the **Assignment Organizer** to assign the **Logic Cell Insertion** logic option to a non-global direct data path connection.

Ignored Logic Cell Insertion logic option on node <*name*> -- illegal value specified for number of logic cells to insert

---

CAUSE: You turned on the **Logic Cell Insertion** logic option to add logic cell buffers to the specified node. However, you specified an illegal value for the number of logic cells to insert. This value must be a number greater than zero and less than 100.

ACTION: Use the **Assignment Organizer** to specify a legal value for the number of logic cells to insert.

Ignored Maximum Fan-Out assignment

CAUSE: You turned on the **Maximum Fan-Out** logic option for one or more nodes in the design. However, the Compiler could not honor this assignment on one or more of the assigned nodes because the assignment was illegal for the node or the logic cell has multiple outputs. As a result, the Compiler ignored the logic option assignment.

ACTION: No action is required. To avoid receiving this message in the future, make sure that you assign the **Maximum Fan-Out** logic option only to nodes that are legal for this assignment.

Ignored Maximum Fan-Out logic option for <*name*>

CAUSE: You turned on the **Maximum Fan-Out** logic option for the specified node and specified a fan-out limit. However, the fan-out from the node exceeds the limit you specified, and the Compiler cannot reduce the fan-out below the specified limit.

ACTION: No action is required. To avoid recieving this message in the future, modify the design so that the fan-out from the node does not exceed the limit.

Ignored Power-Up High logic option setting for state bit <*name*> -- using reset state's zero value instead

---

CAUSE: You assigned the **Power-Up High** logic option to the specified state bit, but the reset state has a value of zero at this state bit. When this occurs, the Logic Synthesizer ignores the **Power-Up High** logic option for the state bit.

ACTION: No action is required.

Ignored Virtual Pin assignment on output enabled pin *<name>*

CAUSE: You assigned the **Virtual Pin** logic option to the specified output enabled pin. However, this assignment is illegal for output enabled pins. As a result, the Compiler ignored the **Virtual Pin** assignment for the pin.

ACTION: No action is required. However, to avoid receiving this message in the future, remove the **Virtual Pin** assignment from the output enabled pin.

Ignored Virtual Pin assignment on registered pin <*name*>

CAUSE: You assigned the **Virtual Pin** logic option to the specified registered pin. However, this assignment is illegal for registered pins. As a result, the Compiler ignored the **Virtual Pin** assignment for the pin.

ACTION: No action is required. However, to avoid receiving this message in the future, remove the **Virtual Pin** assignment from the registered pin.

Ignoring assignments for <*name*>[<*number*>][<*number*>] because it is an invalid assignment target -- <*name*> is a single bit, and cannot have members

---

CAUSE: You assigned the specified signal as a target, but the specified bit is a single bit, and cannot have members. As a result, the MAX+PLUS II Advanced Synthesis software is ignoring the assignment.

ACTION: Modify the assignment so that it contains a valid target.

Ignoring CLKLOCK primitives with ClockBoost parameter of 1 -- ClockLock PLLs not supported in target device family

---

CAUSE:	The design contains `CLKLOCK` primitives with a ClockBoost parameter equal to 1. However, the Compiler is ignoring this parameter because the current target device you specified for the design does not support ClockLock PLLs. This ClockBoost parameter is safely ignored because it is equal to 1.
ACTION:	No action is required.

Ignoring CLKLOCKx1 Input Frequency logic option -- ClockLock PLLs not supported in target device family

---

CAUSE: You assigned the **CLKLOCKx1 Input Frequency** logic option to one or more nodes in the current design. However, the Compiler is ignoring this logic option because the current target device you specified for the design does not support ClockLock PLLs.

ACTION: No action is required. If you want to use a ClockLock PLL, target a device family that supports ClockLock PLLs.

Ignoring setting <*name*> -- value <*name*> is illegal. Refer to --help for legal values.

CAUSE: You started a MAX+PLUS II Advanced Synthesis software command-line command with the specified setting and value. However, the value is illegal or contains a semantic error. The MAX+PLUS II Advanced Synthesis software ignored the illegal setting.

ACTION: No action is required. If you want to specify the setting, make sure the setting is legal, correct any syntax or semantic errors, and run the command again. You may need to recover a previous version of the file to determine the exact code the MAX+PLUS II Advanced Synthesis software excluded. Make sure the settings you add conform to the usage code statement definitions and the format specific to the settings and configuration file section.

Ignoring Virtual Pin assignment made to bidirectional pin *<name>*

CAUSE: You assigned the **Virtual Pin** logic option to the specified bidirectional pin in the design. However, you must assign the **Virtual Pin** logic option only to input or output pins. When this condition occurs, the Compiler ignores the **Virtual Pin** logic option assignment to the bidirectional pin.

ACTION: No action is required. However, to avoid receiving this message in the future, remove the **Virtual Pin** assignment from the bidirectional node.

Ignoring Virtual Pin assignment to node <*name*>

CAUSE: You assigned the **Virtual Pin** logic option to the specified node that is not an input or output pin. However, you must assign the **Virtual Pin** logic option only to input or output pins. When this condition occurs, the Compiler ignores the **Virtual Pin** logic option assignment to the node.

ACTION: No action is required. However, to avoid receiving this message in the future, remove the **Virtual Pin** assignment from the node.

Ignoring Virtual Pin assignment(s) -- current device does not support Virtual Pin assignment

---

CAUSE: You assigned the **Virtual Pin** logic option to one or more nodes in the current design. However, the current target device you specified for the design does not support virtual pins. When this condition occurs, the Compiler ignores the **Virtual Pin** assignment(s). The **Virtual Pin** logic option is only available for APEX 20K and Stratix device families.

ACTION: No action is required. However, to avoid receiving this message in the future, remove the **Virtual Pin** assignment from the design, or target a device family that supports virtual pins.

Illegal database file name <*name*> -- file name is too long

CAUSE: The specified temporary database file name is illegal because the corresponding project name or top-level design entity name from which the database file is generated contains too many characters. The maximum file name length varies according to the operating system, but file names with more than 240 characters, including file name extensions, may be truncated by the operating system, or may cause an operating system error.

ACTION: Rename the project or top-level design entity so the project name or top-level design entity name contains fewer characters. Altera recommends creating project names, design entity names, and file names with a maximum of 80 name characters.

Illegal EDA <*name*> tool name <*name*>

CAUSE: The EDA tool name in the Third Party EDA Tools Section of the Project Settings File (**.psf**) is illegal. This error can occur if you edited the PSF manually or if you set the EDA tool name by using the `project add_assignment` Tcl command. Altera recommends using the MAX+PLUS II Advanced Synthesis software to modify settings, rather than editing settings and configuration files manually.

ACTION: Specify a legal EDA tool name. If you use the `project add_assignment` Tcl command, use a legal EDA tool name.

Implemented <*number*> bidirectional pins

CAUSE: The Logic Synthesizer implemented the specified number of bidirectional pins during logic synthesis.

ACTION: No action is required.

Implemented <*number*> ClockLock PLLs

CAUSE: The Logic Synthesizer implemented the specified number of ClockLock PLLs during logic synthesis.

ACTION: No action is required.

Implemented <*number*> content-addressable memory segments

CAUSE: The Logic Synthesizer implemented the specified number of content-addressable memory (CAM) block segments during logic synthesis.

ACTION: No action is required.

Implemented <*number*> device resources

CAUSE: The Logic Synthesizer implemented the specified number of device resources during logic synthesis. Additional messages list the specific type and number of implemented device resources.

ACTION: No action is required.

Implemented <*number*> DSP elements

CAUSE: The Logic Synthesizer implemented the specified number of DSP elements during logic synthesis. A DSP element is the smallest component of a DSP block and represents a 9x9 multiplier.

ACTION: No action is required.

Implemented <*number*> input pins

CAUSE: The Logic Synthesizer implemented the specified number of input pins during logic synthesis.

ACTION: No action is required.

Implemented <*number*> logic cells

CAUSE: The Logic Synthesizer implemented the specified number of logic cells during logic synthesis.

ACTION: No action is required.

Implemented <*number*> macrocells

CAUSE: The Logic Synthesizer implemented the specified number of macrocells during logic synthesis.

ACTION: No action is required.

Implemented <*number*> output pins

CAUSE: The Logic Synthesizer implemented the specified number of output pins during logic synthesis.

ACTION: No action is required.

Implemented <*number*> RAM segments

CAUSE: The Logic Synthesizer implemented the specified number of RAM segments during logic synthesis.

ACTION: No action is required.

Implemented <*number*> shareable expanders

CAUSE: The Logic Synthesizer implemented the specified number of shareable expanders during logic synthesis.

ACTION: No action is required.

Implemented <*number*> tri-state elements

CAUSE: The Logic Synthesizer implemented the specified number of tri-state elements during logic synthesis.

ACTION: No action is required.

Implementing hierarchy <*name*> using look-up table logic

CAUSE: You selected the **Auto APEX Technology Mapper** option to allow the MAX+PLUS II Advanced Synthesis software to decide how to map the logic of the specified hierarchy. The Logic Synthesizer implemented the hierarchy using look-up table (LUT) logic.

ACTION: If you want to implement the hierarchy in LUTs, no action is required. Otherwise, assign the hierarchy to another device resource using the **APEX Technology Mapper** option

Implementing hierarchy <*name*> using product term logic

CAUSE: You selected the **Auto APEX Technology Mapper** option to allow the MAX+PLUS II Advanced Synthesis software to decide how to map the logic of the specified hierarchy. The Logic Synthesizer implemented the hierarchy using product term logic.

ACTION: No action is required.

In current operating mode, WYSIWYG RAM primitive <*name*> cannot have <*name*> port or parameter connected

---

CAUSE: The specified WYSIWYG RAM primitive, when in its current operating mode, has the specified port or parameter connected. The WYSIWYG primitive cannot have the specified port or parameter connected when in its current operating mode. The specified port or parameter must be disconnected.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

In single-port mode, WYSIWYG RAM primitive <*name*> cannot have different raddr and waddr ports

---

CAUSE: The specified WYSIWYG RAM primitive, which is in single-port mode, is using two different signals for the `raddr` and `waddr` ports. The `raddr` and `waddr` ports must be connected to the same source or the WYSIWYG RAM primitive must be changed to dual-port mode.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

In the current operating mode, WYSIWYG RAM primitive <*name*> cannot have <*name*> port or parameter specified on port B

CAUSE: The specified WYSIWYG RAM primitive, when in its current operating mode, has the specified port or parameter specified on port B. The WYSIWYG RAM primitive can only have an input, output, or parameter specified on port B if the RAM is in bidir, dual-port, or quad-port mode.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

Input port other than cin or cascin of WYSIWYG LCELL primitive <*name*> is fed by another WYSIWYG LCELL primitive's cout or cascout port

---

CAUSE: The `cout` or `cascout` port of a WYSIWYG LCELL primitive connects to the input port other than `cin` or `cascin` of the specified WYSIWYG LCELL primitive. The port connection must be removed, or the `cin` or `cascin` of the specified WYSIWYG LCELL primitive must be connected to a `cin` or `cascin` port.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

Inserted <*number*> logic cells after node <*name*>

CAUSE: You turned on the **Logic Cell Insertion** logic option as a single point assignment for the specified node. As a result, the Compiler inserted the specified number of logic cell buffers after the specified node.

ACTION: No action is required.

Inserted <*number*> logic cells between source node <*name*> and destination node <*name*>

CAUSE: You used a point-to-point **Logic Cell Insertion** logic option assignment to direct the Compiler to insert the specified number of logic cell buffers between the specified source and destination nodes. As a result, the Compiler is reporting that it successfully inserted the logic cell buffers.

ACTION: No action is required.

Inserted <*number*> logic cells for Maximum Fan-Out assignment on <*name*>

CAUSE: You turned on the **Maximum Fan-Out** logic option for the specified node. However, the fan-out from the node exceeds the limit you specified. Therefore, the Compiler inserted the specified number of logic cells so the fan-out count is below the specified limit.

ACTION: No action is required.

Invalid port name <*name*>

CAUSE: The MAX+PLUS II Advanced Synthesis software could not map the specified port name of an EDA tool function to the appropriate MAX+PLUS II function name.

ACTION: Make sure you have selected the appropriate EDA tool for design entry or synthesis, and the appropriate Library Mapping File (.**lmf**). In addition, you should make sure that all pairs of `FUNCTION` and `RETURN` statements in the LMF have a one_to_one correspondence.

Keyword <*name*> in settings and configuration file <*name*> is not a legal keyword of section <*name*> -- deleting illegal keyword

---

CAUSE: You added the specified keyword to the specified section in the specified settings and configuration file, but the specified keyword is not a part the specified section, or there is a semantic error in the keyword name. The MAX+PLUS II Advanced Synthesis software rewrote the file to exclude the illegal keyword.

ACTION: If you added the illegal keyword in error, and do not want to recover the settings you specified, then no action is required. If you want to specify the deleted keyword in the file, make sure the keyword is legal for the specified section and correct any syntax or semantic errors in the keyword, then add the setting to the file again. You may need to recover a previous version of the file to determine the exact code the MAX+PLUS II Advanced Synthesis software excluded. Make sure the settings you add conform to the usage code statement definitions and the format specific to the settings and configuration file section. To avoid receiving this message in the future, Altera recommends using the MAX+PLUS II Advanced Synthesis software dialog boxes and wizards to modify settings, rather than editing the settings and configuration files manually.

LATCH primitive <*name*> is permanently disabled

CAUSE: In a design file in the current design, you connected the `ena` port of the specified `LATCH` primitive to `GND`. As a result of logic synthesis during compilation, the `LATCH` primitive is permanently disabled.

ACTION: No action is required. Check the design file for errors. If you do not want the `LATCH` primitive to be permanently disabled, connect the `ena` port to a signal other than `GND`.

LATCH primitive <*name*> is permanently enabled

CAUSE: In a design file in the current design, you connected the `ena` port of the specified `LATCH` primitive to `VCC`. As a result of logic synthesis during compilation, the `LATCH` primitive is reduced to an `LCELL` buffer, and is permanently enabled.

ACTION: No action is required. Check the design file for errors. If you do not want the `LATCH` primitive to be permanently enabled, connect the `ena` port to a signal other than `VCC` to control the latch, or substitute an `LCELL` buffer for the latch.

Library Mapping File <*name*> contains one ore more syntax errors

---

CAUSE: You attempted to compile a file generated by another EDA tool, but the Library Mapping File (.**lmf**) used to map the EDA tool functions to MAX+PLUS II functions contains syntax errors. This error may occur if you edited an Altera-provided LMF. As a result, the Compiler cannot map the appropriate functions.

ACTION: Make sure the LMF file does not contain syntax errors, or specify the appropriate Altera-provided LMF for the design entry or synthesis tool you selected, if it has not been modified.

License for core <*name*>, version <*version*> is expired

CAUSE: You instantiated the specified core and attempted to compile the design. However, the license for the core is expired. As a result, the MAX+PLUS II Advanced Synthesis software cannot compile the design.

ACTION: Contact the core vendor for information on purchasing a new license.

LMF mapping record <*name*> -> <*name*> missing the <*name*> port mapping(s) for node <*name*>

CAUSE: The MAX+PLUS II Advanced Synthesis software could not map the specified port because the port is missing the specified port mapping for the specified node.

ACTION: Make sure you have selected the appropriate EDA tool for design entry or synthesis, and the appropriate Library Mapping File (.**lmf**). In addition, you should make sure that all pairs of `FUNCTION` and `RETURN` statements in the LMF have a one_to_one correspondence.

Logic between pin <*name*> and its register contains a loop -- can't implement Power-Up High option

---

CAUSE: You turned on the **Power-Up High** option for the I/O register of the specified pin. However, the Logic Synthesizer encountered a loop between the pin and the register, and is unable to implement the **Power-Up High** option.

ACTION: Remove the looped logic in the design, or turn off the **Power-Up High** option.

Logic cell <*name*>

CAUSE: The specified node is a redundant logic cell, as indicated by the top-level message that precedes this submessage.

ACTION: No action is required.

Logical RAM <*name*> and logical RAM <*name*> use the same logical RAM name <*name*>

CAUSE: The two specified logical RAM blocks use the same specified name. However, all logical RAMs must have unique names.

ACTION: Modify the design so that all logical RAMs have unique names.

LUT_MASK value <*number*> must be four-digit hexadecimal number

---

CAUSE: The specified `LUT_MASK` value in the WYSIWYG LCELL primitive is not in the format of a four-digit hexadecimal number.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

MAX+PLUS II Advanced Synthesis software applied gate-level register retiming to <*number*> clock domains

---

CAUSE: You directed the MAX+PLUS II Advanced Synthesis software to perform gate-level register retiming. As a result, the MAX+PLUS II Advanced Synthesis software applied gate-level retiming to the specified clock domains, created new registers, and removed some registers. For more details, see the Gate-Level Retiming section of the Report window.

ACTION: No action is required.

MAX+PLUS II Advanced Synthesis software applied gate-level register retiming to clock <*clock name*>: created <*number*> new registers, removed <*number*> registers, left <*number*> registers untouched

---

CAUSE: You directed the MAX+PLUS II Advanced Synthesis software to perform gate-level register retiming. As a result, the MAX+PLUS II Advanced Synthesis software applied gate-level retiming to the specified clock, created the specified number of new registers, removed the specified number of registers, and left the specified number of registers untouched. For more details, see the Gate-Level Retiming section of the Report window.

ACTION: No action is required.

MAX+PLUS II Advanced Synthesis software is resynthesizing <*number*> WYSIWYG logic cells using the <*name*> technology mapper and leaves <*number*> WYSIWYG logic cells untouched

---

CAUSE: You directed the MAX+PLUS II Advanced Synthesis software to optimize netlists by unmapping WYSIWYG primitives during synthesis. As a result, the MAX+PLUS II Advanced Synthesis software is resynthesizing the specified number of lcells, and leaves the specified number of lcells untouched. The MAX+PLUS II Advanced Synthesis software does not touch arithmetic lcells or any WYSIWYG lcells that have the option 'Netlist Optimizations' set to 'Never Allow'.

ACTION: No action is required.

Memory Initialization File or Hexadecimal (Intel-Format) File for RAM is not specified -- setting initial contents to 0

CAUSE: You did not specify a file name for the Memory Initialization File (**.mif**) or the Hexadecimal (Intel-Format) File (**.hex**), or you set the name to `NONE`. The MAX+PLUS II Advanced Synthesis Compiler is therefore setting all initial values to 0.

ACTION: No action is required. If you do not want all initial values set to 0, make sure the MIF or HEX File exists. Do not include empty strings as file names.

Net <*name*> cannot be assigned more than one value

CAUSE: In the Continuous Assignments section of a Verilog Design File (**.v**), you assigned more than one value to the specified net. You can assign only one value to (that is, use only one Continuous Assignment Statement for) a net in a Verilog Design File.

ACTION: Make sure the Verilog Design File contains only one Continuous Assignment Statement that assigns a value to the signal.

Net is fed by <*name*>

CAUSE: In the Continuous Assignments section of a Verilog Design File (**.v**), you assigned more than one value to the specified net. You can assign only one value to (that is, use only one Continuous Assignment Statement for) a net in a Verilog Design File.

ACTION: Make sure the Verilog Design File contains only one Continuous Assignment Statement that assigns a value to the signal.

No clock transition on <*name*> register

CAUSE: The specified register's clock is stuck at `VCC` or `GND`, and the Compiler reduces the register to its power-up state. You may have specified this condition in the design file, or the condition may be the result of logic synthesis.

ACTION: If you expected this result, no action is required. Otherwise, check the design file for errors. Or, you can add a preset and/or clear input to toggle the register.

No output dependent on input pin <*name*>

CAUSE: The specified input pin does not drive any logic. This message occurs when no outputs are dependent on the specified input pin.

ACTION: If this condition is intentional, no action is required. If this is condition unintentional, make sure all input pins in the design drive logic.

Node <*name*>

CAUSE: The Compiler either found or removed the specified redundant node during compilation. This message is a sub-message of the message that precedes it in the Messages window and in the Messages section of the Report window.

ACTION: No action is required.

Node <*name*>

CAUSE: The specified node is part of a top-level message in the hierarchy tree that appears above this message in the Messages window.

ACTION: No action is required.

Node <*name*>

CAUSE: The specified node is part of another message that appears above this message in the Messages window.

ACTION: No action is required.

Node <*name*> in settings and configuration file <*name*> is illegal -- deleting illegal node

---

CAUSE: You added the specified node to the specified settings and configuration file, but the MAX+PLUS II Advanced Synthesis software detects a semantic error in the node setting, or the node does not exist. The MAX+PLUS II Advanced Synthesis software rewrote the file to exclude the illegal node setting.

ACTION: If you added the illegal node in error, and do not want to recover the settings you specified, then no action is required. If you want to specify the node, make sure the node setting is legal in the specified file and correct any syntax or semantic errors in the node name, then add the setting to the file again. You may need to recover a previous version of the file to determine the exact code the MAX+PLUS II Advanced Synthesis software excluded. Make sure the settings you add conform to the usage code statement definitions and the format specific to the settings and configuration file section. To avoid receiving this message in the future, Altera recommends using the MAX+PLUS II Advanced Synthesis software dialog boxes and wizards to modify settings, rather than editing the settings and configuration files manually.

Node <*name*> is missing source

CAUSE: A node is connected to one or more destinations but does not have a source. This error can also occur if you have not correctly named a node that is connected to a bus in a Graphic Design File (**.gdf**) or Block Design File (**.bdf**).

ACTION: Connect the node to the output of a logic function. If you want to use the default unconnected input to a macrofunction, make sure that nothing is connected to the input of the macrofunction.

Node <*name*> of type <*type*>

CAUSE: The specified node is part of a top-level message in the hierarchy tree that appears above this message in the Messages window.

ACTION: No action is required.

Node instance <*name*> instantiates undefined entity <*name*>

CAUSE: You attempted to compile the current design, but the Database Builder cannot compile the design because the specified node instantiates the specified undefined entity.

ACTION: Define the specified entity or change the calling entity. Also, make sure that the required user library paths are specified correctly. If the project contains EDIF Input Files (**.edf**), verify that you specified the EDA synthesis tool settings correctly. Verilog HDL Design File (**.v**) and Verilog Quartus Mapping File (**.vqm**) entity names are case sensitive.

Node instance <*name*> must instantiate Text Design File entity <*name*> with named parameters

---

CAUSE: The specified node instance instantiates the specified Text Design File (**.tdf**) entity without named parameters. However, you must instantiate the TDF entity with named parameters.

ACTION: Instantiate the TDF entity with named parameters.

Node name <*name*> used by assignment of type <*type*> with value <*text*> is illegal

---

CAUSE: An assignment in a settings and configuration file with the specified type and value refers to the specified node name, but the node name contains illegal name characters. The MAX+PLUS II Advanced Synthesis software ignored and deleted the assignment.

ACTION: Create another assignment with a node name containing legal name characters.

Not allowed to move <*number*> registers because of timing assignments

---

CAUSE: You directed the MAX+PLUS II Advanced Synthesis software to perform gate-level register retiming. However, the MAX+PLUS II Advanced Synthesis software cannot move the specified number of registers because they are involved in timing assignments.

ACTION: If you want to allow the MAX+PLUS II Advanced Synthesis software to move these registers, set the **Netlist Optimizations** logic option to **Always Allow**. The retiming algorithm will not change the timing assignments due to the movement of these registers and it may merge registers that have incompatible timing assignments. Otherwise, no action is required.

Not allowed to move <*number*> registers because of user assignments

---

CAUSE: You directed the MAX+PLUS II Advanced Synthesis software to perform gate-level register retiming. However, the MAX+PLUS II Advanced Synthesis software cannot move the specified number of registers because either you set the **Netlist Optimizations** logic option to **Never allow** or your set the Preserve Register logic option to on for these registers.

ACTION: If you want to allow the MAX+PLUS II Advanced Synthesis software to move these registers, set the **Netlist Optimizations** logic option to **Always Allow**, or set the Preserve Register logic option to off. Otherwise, no action is required.

Not allowed to move <*number*> registers because they are connected to SERDES

CAUSE: You directed the MAX+PLUS II Advanced Synthesis software to perform gate-level register retiming. However, the MAX+PLUS II Advanced Synthesis software cannot move the specified number of registers because they are fed by the data-out port on a SERDES receiver, or a data-in port on a SERDES transmitter.

ACTION: If you want to allow the MAX+PLUS II Advanced Synthesis software to move these registers, set the **Netlist Optimizations** logic option to **Always Allow**. Otherwise, no action is required.

Not allowed to move <*number*> registers because they are directly fed by input pins

---

CAUSE: You directed the MAX+PLUS II Advanced Synthesis software to perform gate-level register retiming. However, the MAX+PLUS II Advanced Synthesis software cannot move the specified number of registers because they are directly fed by input pins.

ACTION: If you want to allow the MAX+PLUS II Advanced Synthesis software to move these registers, set the **Netlist Optimizations** logic option to **Always Allow**. Otherwise, no action is required.

Not allowed to move <*number*> registers because they are fed by input pins

---

CAUSE: You directed the MAX+PLUS II Advanced Synthesis software to perform gate-level register retiming. However, the MAX+PLUS II Advanced Synthesis software cannot move the specified number of registers because they are fed by pins directly or through combinational logic. Some of these registers are not moved because you turned off the option **Allow register retiming to trade off Tsu/Tco with Fmax**. Others are not moved because they are directly fed by input pins.

ACTION: If you want to allow the MAX+PLUS II Advanced Synthesis software to move these registers, turn on **Allow register retiming to trade off Tsu/Tco with Fmax**, or set the **Netlist Optimizations** logic option to **Always Allow**. Otherwise, no action is required.

Not allowed to move <*number*> registers because they are fed by registers in a different clock domain

---

CAUSE: You directed the MAX+PLUS II Advanced Synthesis software to perform gate-level register retiming. However, the MAX+PLUS II Advanced Synthesis software cannot move the specified number of registers because they are fed by registers in a different clock domain, and may be synchronization registers.

ACTION: If you want to allow the MAX+PLUS II Advanced Synthesis software to move these registers, set the **Netlist Optimizations** logic option to **Always Allow**. Otherwise, no action is required.

Not allowed to move <*number*> registers because they feed clock or asynchronous control signals of other registers

---

CAUSE: You directed the MAX+PLUS II Advanced Synthesis software to perform gate-level register retiming. However, the MAX+PLUS II Advanced Synthesis software cannot move the specified number of registers because they feed clock or asynchronous control signals of another register.

ACTION: If you want to allow the MAX+PLUS II Advanced Synthesis software to move these registers, set the **Netlist Optimizations** logic option to **Always Allow**. Otherwise, no action is required.

Not allowed to move <*number*> registers because they feed output pins

---

CAUSE: You directed the MAX+PLUS II Advanced Synthesis software to perform gate-level register retiming. However, the MAX+PLUS II Advanced Synthesis software cannot move the specified number of registers because they feed output pins directly or through combinational logic. Some of these registers are not moved because you turned off the option **Allow register retiming to trade off Tsu/Tco with fmax**. Others are not moved because they directly feed output pins.

ACTION: If you want to allow the MAX+PLUS II Advanced Synthesis software to move these registers, turn on **Allow register retiming to trade off Tsu/Tco with fmax**, or set the **Netlist Optimizations** logic option to **Always Allow**. Otherwise, no action is required.

Not allowed to move <*number*> registers because they feed output pins directly

---

CAUSE: You directed the MAX+PLUS II Advanced Synthesis software to perform gate-level register retiming. However, the MAX+PLUS II Advanced Synthesis software cannot move the specified number of registers because they feed output pins directly.

ACTION: If you want to allow the MAX+PLUS II Advanced Synthesis software to move these registers, set the **Netlist Optimizations** logic option to **Always Allow**. Otherwise, no action is required.

Not allowed to move <*number*> registers because they feed registers in a different clock domain

---

CAUSE: You directed the MAX+PLUS II Advanced Synthesis software to perform gate-level register retiming. However, the MAX+PLUS II Advanced Synthesis software cannot move the specified number of registers because they feed the data, clock enable, or synchronous control signals of registers in a different clock domain.

ACTION: If you want to allow the MAX+PLUS II Advanced Synthesis software to move these registers, set the **Netlist Optimizations** logic option to **Always Allow**. Otherwise, no action is required.

Not allowed to move registers -- <*number*> registers affected

---

CAUSE: You directed the MAX+PLUS II Advanced Synthesis software to perform gate-level register retiming. However, the MAX+PLUS II Advanced Synthesis software cannot move the specified number of registers.

ACTION: If you want to allow the MAX+PLUS II Advanced Synthesis software to move these registers, set the **Netlist Optimizations** logic option to **Always Allow**. Otherwise, no action is required.

OpenCore Plus time-limited core <*name*> may be used for hardware evaluation only

---

CAUSE: You instantiated the specified OpenCore Plus time-limited core to your project. This core may be used for hardware evaluation only.

ACTION: You will be able to program an Altera device only with a download cable through the MAX+PLUS II software, and your design will run for only a given number of clock cycles. See core documentation for clock cycle limits and disable behavior, or see the OpenCore Plus Hardware Evaluation of MegaCore Functions application note, which is available at the Altera web site for general information. To compile the non-time-limited version of this core, change your User Libraries settings by removing "lib_time_limited" from your core, and adding "lib". You will also need to reinstantiate the core in your design.

Out of memory

CAUSE: REDIRECT EVHDX_AZ_NA_OUTOFMEMORY

ACTION: REDIRECT EVHDX_AZ_NA_OUTOFMEMORY

Output pins are stuck at VCC or GND

CAUSE: Output pins minimized to `VCC` or `GND` in a design file. This condition may be the result of optimizations performed during logic synthesis.

ACTION: If you intended the output pins to behave in this manner, no action is required. Otherwise, check the design file for errors and ensure that the project's logic does not reduce to `VCC` or `GND`.

Parallel expander chain length <*number*> exceeds limit of current device. Change length to <*number*>

CAUSE: You indicated the specified value for the **Parallel Expander Chain Length** logic option. However, the parallel expander chain length value exceeds the specified maximum allowable limit for the current target device.

ACTION: Change the **Parallel Expander Chain Length** value to the specified maximum value, or change the current target device to a device that supports the specified chain length.

Parameter <*name*> is not supported for WYSIWYG primitive <*name*>

CAUSE: The specified WYSIWYG primitive is using the specified parameter; however, the specified parameter is not supported for the WYSIWYG primitive.

ACTION: Modify the design to remove the parameter value for the WYSIWYG primitive. If the problem persists, contact Altera Applications for assistance.

Performing gate-level register retiming

CAUSE: You directed the MAX+PLUS II Advanced Synthesis software to perform gate-level register retiming during synthesis to try to get improved $\mathbf{f_{MAX}}$ results. As a result, the MAX+PLUS II Advanced Synthesis software may remove some registers and create some new registers.

ACTION: No action is required.

Performing hierarchical synthesis which affects fitting and performance

---

CAUSE: You assigned the **Preserve Hierarchical Boundary** logic option to one or more subdesigns in the current project, or assigned different values across subdesigns for some entity-specific logic options, which caused the Compiler to infer hierarchical boundaries.

ACTION: No action is required. To avoid receiving this message in the future, you can remove the hierarchical assignments, and make sure entity-specific assignments have the same value across the hierarchies.

Pin <*name*> has no register for Power-Up High option

CAUSE: You assigned the **Power-Up High** option to the specified input, output, or bidirectional pin; however, the Compiler could not find an I/O register associated with the pin. This option only functions correctly when it is assigned to an input pin that drives the data input of an I/O register, or to an output or bidirectional pin driven an I/O register.

ACTION: If you want to use the **Power-Up High** option, remove any extra logic between the specified pin and its register, other than optional `TRI` or `OPNDRN` buffers. If you do not need to use the **Power-Up High** option, delete the **Power-Up High** option to avoid receiving this message in the future.

Pin <*name*> is virtual input pin

CAUSE: You assigned the **Virtual Pin** logic option to the specified input pin in the design.

ACTION: No action is required.

Pin <*name*> is virtual output pin

CAUSE: You assigned the **Virtual Pin** logic option to the specified output pin in the design.

ACTION: No action is required.

Pin <*name*> stuck at <*value*>

CAUSE: The specified output pin is stuck at `VCC` or `GND`. This condition may be the result of optimizations performed during logic synthesis.

ACTION: If you intended the output pins to behave in this manner, no action is required. Otherwise, check the design file for errors and ensure that the project's logic does not reduce to this vlaue.

PLL <*name*> uses clock settings parameter <*name*> -- clock settings are no longer supported for PLLs

---

CAUSE: The specified WYSIWYG PLL primitive uses the specified clock settings parameter. However, clock settings parameters are no longer supported for WYSIWYG PLL primitives.

ACTION: Modify the design to remove the clock settings parameters from the WYSIWYG PLL primitive.

Port <*name*> does not exist in macrofunction <*name*>

CAUSE: You connected the specified macrofunction to a lower-level macrofunction through the specified port that does not exist. As a result, the MAX+PLUS II Advanced Synthesis software cannot compile the design.

ACTION: Remove the illegal connection or create a port for the lower-level macrofunction.

Port <*name*> does not exist in macrofunction <*name*>. The range of the port is wider in the macrofunction than in the design.

CAUSE:	The specified port does not exist in the specified macrofunction. Pins of the same name exist in the design, but the range of the port is wider in the macrofunction than in the design.
ACTION:	Modify the macrofunction so that the width of the port in the macrofunction matches the width of the port in the design.

Port <*name*> does not exist in primitive <*name*> of instance <*name*>

CAUSE: You connected a port to the specified primitive through the specified port name, but the name does not exist.

ACTION: Make sure that you spelled the port name correctly and replace the port name with a legal primitive port name.

Port <*name*> in macrofunction <*name*> has no range declared -- MAX+PLUS II Advanced Synthesis software will connect the port to pin <*name*> because the pin is a member of a single bit bus with the same name as the port

---

CAUSE: You connected the specified port in the specified macrofunction, but you did not declare a range for the port. As a result, the MAX+PLUS II Advanced Synthesis software will connect the port to pin %2!s! because the pin is a member of a single bit bus with the same name as the port.

ACTION: No action is required. If this is not what you intended and you created the file in the MAX+PLUS II Advanced Synthesis software, modify the macrofunction so that the range of the port in the macrofunction matches the range of the port in the design.

Port at position <*number*> does not exist in macrofunction <*name*>

CAUSE: You connected the specified macrofunction to a lower-level macrofunction through the specified port that does not exist. As a result, the MAX+PLUS II Advanced Synthesis software cannot compile the design.

ACTION: Remove the illegal connection or create a port for the lower-level macrofunction.

Port direction mismatch for entity <*name*> at port <*name*>. Upper entity is expecting <*name*> pin while lower entity is using <*name*> pin.

CAUSE: You connected the specified upper entity at the specified port in a different direction than the direction of the specified lower entity pin. As a result, the MAX+PLUS II Advanced Synthesis software cannot compile the design.

ACTION: Remove the illegal connection or connect the upper entity in the same direction as the lower entity pin.

Power-Up High option on register <*name*> will override Power-Up High option on pin <*name*>

CAUSE: You turned on the **Power-Up High** option for both the specified pin and its register. However, these options conflict with each other due to an inversion between the pin and the register. The Logic Synthesizer is overriding the option on the pin with the option on the register.

ACTION: If you want the **Power-Up High** option on the register to override the **Power-Up High** option on the pin, no action is required. Otherwise, remove any inversion between the pin and register that could affect power-up.

Power-up level of register <*name*> is not specified -- using unspecified power-up level

---

CAUSE: You did not specify the power-up state of the specified register. The Logic Synthesizer is using a nondefault setting for the power-up state so it can minimize the delays or resources in the design.

ACTION: No action is required. If you want to specify a particular power-up state, you can add a register assignment to force that power-up state. If you want to specify that the Logic Synthesizer should not use a nondefault setting for unspecified power-up states in order to minimize delays or design resources, set the `ALLOW_POWER_UP_DONT_CARE` option to `OFF` in the Project Settings File (**.psf**).

Presettable registers will power up high

CAUSE: Your design contains registers that are fed by a preset signal but not by a clear signal. When this condition occurs, the MAX+PLUS II Advanced Synthesis software forces these registers to power up high.

ACTION: No action is required. However, if you want these registers to power up low, turn the **Power up High** option off for these registers.

Primitive <*name*> of type <*type*> cannot feed CASCADE primitive <*name*> because the primitive already contains fan-out to one or more CASCADE primitives

---

CAUSE: The specified primitive cannot fan out to the specified `CASCADE` primitive because it fans out to one or more `CASCADE` primitives.

ACTION: Modify the design so that the specified primitive fans out to only one `CASCADE` primitive by removing all but one `CASCADE` output from the specified primitive and deleting the remaining `CASCADE` primitives.

Project Setting File <*name*> is read-only and cannot be edited

---

CAUSE: You tried to edit the specified Project Setting File (**.psf**) using the following pages or dialog box, but the PSF is read-only:

- The **Assignment Organizer** dialog box
- The **Default Logic Option Settings** page
- The **Default Parameter Settings** page
- The **EDA Tools Settings** page
- The pages under the **Files & Directories** page
- The pages under the **HDL Input Settings** page
- The pages under the **Timing Settings** page

ACTION: Remove the read-only attribute of the PSF.

Project too complex: hierarchy path is too long

CAUSE: The depth of the design hierarchy is too deep for the MAX+PLUS II Advanced Synthesis software to process. This could also be due to an undetected recursion.

ACTION: Reduce the depth of the hierarchies. If necessary, modify the design so that it does not contain any recursive logic.

raddr port is not connected in WYSIWYG RAM primitive *<name>*

CAUSE: The specified WYSIWYG RAM primitive does not have a connected `raddr` port. The `raddr` port must be connected.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

RAM ADDRESS_WIDTH parameter specified for WYSIWYG primitive <*name*> at port A must operate in deep RAM mode

---

CAUSE: The specified WYSIWYG primitive with the RAM `ADDRESS_WIDTH` specified at port A is not operating in deep RAM mode. The operating mode must be changed to deep RAM mode.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

RAM ADDRESS_WIDTH parameter specified for WYSIWYG primitive <*name*> at port B must operate in deep RAM mode

---

CAUSE: The specified WYSIWYG primitive with the RAM `ADDRESS_WIDTH` parameter specified at port B is not operating in deep RAM mode. The operating mode must be changed to deep RAM mode.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

RAM name is <*name*>

CAUSE: You implemented a logic function -- a RAM megafunction (`lpm_ram_dp` or an `lpm_ram_io` megafunction) -- that requires an initial memory content file, specifically, a Memory Initialization File (.**mif**) or Hexadecimal (Intel-Format) File (**.hex**), which is needed to initialize the `lpm_ram_dp` or `lpm_ram_io` megafunction. However, the MAX+PLUS II Advanced Synthesis Compiler cannot read the MIF or HEX File because either the file does not exist or you did not specify its correct directory path. The MAX+PLUS II Advanced Synthesis Compiler is therefore setting all initial values to 0.

ACTION: No action is required. If you do not want all initial values set to 0, make sure the MIF or HEX File exists. Do not include empty strings as file names.

Recursive logic detected while extracting entities

CAUSE: You attempted to compile the current project. However, the MAX+PLUS II Advanced Synthesis software detected an entity that instantiates itself. As a result, the netlist extraction was unsuccessful.

ACTION: Modify the design so that it does not contain any recursive logic and recompile the project.

Reduced register *<name>* with stuck *<name>* port to stuck value *<number>*

CAUSE: The Compiler reduced the specified register to VCC, GND, or a wire because it reduced the inputs to the register to one of the following combinations, in descending order of priority. You may have specified this condition in the design file, or the condition may be a result of logic synthesis.

Active-Low Clear:	Active-Low Preset:	Active-High Aload:	Active-High Clock Enable:	Active-High Clock:	Power-up State:	D_in:	Register Reduces To:
GND							GND
VCC	GND						VCC
	GND						!Clear
VCC	VCC	VCC					Adata
VCC	VCC	GND	GND				Power-up state
VCC	VCC	GND		VCC or GND			Power-up state
	VCC	GND		VCC or GND	Low		GND
VCC		GND		VCC or GND	High		VCC
	VCC	GND	GND		Low		GND
VCC		GND	GND		High		VCC
	VCC	GND			Low	GND	GND
VCC		GND			High	VCC	VCC

ACTION: If you intended the register to behave in this manner, no action is required. Otherwise, check the design file for errors and make sure the project's logic does not reduce to any of these combinations.

☞ The Compiler reduces JK, SR, and T registers to D registers during logic synthesis.

Reduced register <*name*> with stuck signal <*name*> to <*name*>

CAUSE: The specified register is stuck at `VCC` or `GND`. As a result, the Compiler reduced the register to the specified signal.

ACTION: If you intended the register to behave in this manner, no action is required. Otherwise, check the design file for errors and ensure that the project's logic does not reduce to `VCC` or `GND`.

Reduced registers with stuck control signals

CAUSE: A register in the design file are stuck at `VCC` or `GND`. As a result, the Compiler reduced this register.

ACTION: If you intended the register to behave in this manner, no action is required. Otherwise, check the design file for errors and ensure that the project's logic does not reduce to `VCC` or `GND`

Registers with preset signals will power up high

CAUSE: In a project, you connected registers to a preset signal but not a clear signal. As a result, the registers will power up high.

ACTION: No action is required. However, if you want the registers to power up low, turn off the `NOT` Gate Push-Back option on the registers.

Registers with preset signals will power up high

CAUSE: In a project, you connected registers to a preset signal but not a clear signal. As a result, the registers will power up high.

ACTION: No action is required. However, if you want the registers to power up low, turn off the `NOT` Gate Push-Back option on the registers.

Removed the following redundant logic cells

CAUSE: You turned on the **Remove redundant user cells** logic option, and compiled the project. The Compiler found nodes that are functional duplicates of other nodes and removed the nodes in the current design.

ACTION: No action is required. Refer to the submessage(s) in the **Messages** window to determine which redundant logic cells were removed. To prevent the MAX+PLUS II Advanced Synthesis software from removing redundant logic cells, turn off the **Remove Redundant Logic Cells** option.

Removed the following redundant logic cells

CAUSE: You turned on the **Remove Redundant Logic Cells** option. As a result, the MAX+PLUS II Advanced Synthesis software removed the specified redundant logic cells from the design.

ACTION: No action is required. Refer to the submessage(s) in the Message window to determine which redundant logic cells were removed. To prevent the MAX+PLUS II Advanced Synthesis software from removing redundant logic cells, turn off the **Remove Redundant Logic Cells** option.

Removed unnecessary TRI buffers in design files -- buffers are permanently enabled

---

CAUSE: One or more design files in the current design contains `TRI` buffers that have a `VCC` output enable. As a result, the buffers are permanently enabled and the Compiler has removed these unnecessary buffers.

ACTION: No action is required. To see which buffer instances the Compiler removed, locate the source of the message.

Removing OPNDRN node <*name*> that feeds logic

CAUSE: The Compiler removed the specified `OPNDRN` node.

ACTION: No action is required.

Section <*name*> in settings and configuration file <*name*> already contains keyword <*name*> -- deleting duplicate keyword

---

CAUSE: You added the specified keyword to the specified section of the specified settings and configuration file, but the specified keyword already exists in the file. The MAX+PLUS II Advanced Synthesis software rewrote the file to exclude the duplicate keyword.

ACTION: If you added the duplicate keyword in error, and do not want to recover the settings you specified, then no action is required. If you want to specify the deleted keyword in the file, add the settings for the keyword to the keyword that the MAX+PLUS II Advanced Synthesis software did not delete from the file. You may need to recover a previous version of the file to determine the exact code the MAX+PLUS II Advanced Synthesis software excluded. Make sure the settings you add conform to the usage code statement definitions and the format specific to the settings and configuration file section. To avoid receiving this message in the future, Altera recommends using the MAX+PLUS II Advanced Synthesis software dialog boxes and wizards to modify settings, rather than editing the settings and configuration files manually.

Section <*name*> in settings and configuration file <*name*> does not require section keyword <*name*> -- deleting section

---

CAUSE: You added the specified section to the specified settings and configuration file, but you added an unnecessary section keyword. The MAX+PLUS II Advanced Synthesis software rewrote the file to exclude the specified section.

ACTION: If you added the section in error, and do not want to recover the settings you specified in the section, then no action is required. If you want to specify the deleted section in the file, remove the specified section keyword, make sure the section is legal in the specified file, and correct any syntax or semantic errors, then add the section to the file again. You may need to recover a previous version of the file to determine the exact code the MAX+PLUS II Advanced Synthesis software excluded. Make sure the settings you add conform to the usage code statement definitions and the format specific to the settings and configuration file section. To avoid receiving this message in the future, Altera recommends using the MAX+PLUS II Advanced Synthesis software dialog boxes and wizards to modify settings, rather than editing the settings and configuration files manually.

Section <*name*> in settings and configuration file <*name*> requires a section keyword -- deleting section

---

CAUSE: You added the specified section to the specified settings and configuration file, but you did not add the necessary section keyword. The MAX+PLUS II Advanced Synthesis software rewrote the file to exclude the specified section.

ACTION: If you added the section in error, and do not want to recover the settings you specified in the section, then no action is required. If you want to specify the deleted section in the file, add the necessary section keyword, make sure the section and section keyword are legal in the specified file, and correct any syntax or semantic errors, then add the section to the file again. You may need to recover a previous version of the file to determine the exact code the MAX+PLUS II Advanced Synthesis software excluded. Make sure the settings you add conform to the usage code statement definitions and the format specific to the settings and configuration file section. To avoid receiving this message in the future, Altera recommends using the MAX+PLUS II Advanced Synthesis software dialog boxes and wizards to modify settings, rather than editing the settings and configuration files manually.

Section <*name*> not a legal section of settings and configuration file <*name*> -- deleting illegal section

---

CAUSE: You added the specified section to the specified settings and configuration file, but the file cannot contain this section. The MAX+PLUS II Advanced Synthesis software rewrote the file to exclude the illegal section.

ACTION: If you added the illegal section in error, and do not want to recover the settings you specified in the illegal section, then no action is required. If you want to specify the deleted section in the file, make sure the section is a legal section for the specified file and correct any syntax or semantic errors in the section name, then add the section to the file again. You may need to recover a previous version of the file to determine the exact code the MAX+PLUS II Advanced Synthesis software excluded. Make sure the settings you add conform to the usage code statement definitions and the format specific to the settings and configuration file section. To avoid receiving this message in the future, Altera recommends using the MAX+PLUS II Advanced Synthesis software dialog boxes and wizards to modify settings, rather than editing the settings and configuration files manually.

Selected <*name*> state machine encoding method for state machine <*name*>

CAUSE: You set the default **State Machine Processing** logic option to **Auto**. When you set the logic option to **Auto**, the Logic Synthesizer determines the best method to encode the state machine. The Logic Synthesizer is reporting that it selected the specified encoding method for the specified state machine.

ACTION: No action is required. Alternatively, you can specify another state machine processor.

Setting <*name*> in settings and configuration file <*name*> is illegal -- deleting illegal setting

---

CAUSE: You added the specified setting to the specified settings and configuration file, but the setting is illegal or the MAX+PLUS II Advanced Synthesis software detects a semantic error in the setting name. The MAX+PLUS II Advanced Synthesis software rewrote the file to exclude the illegal setting.

ACTION: If you added the illegal setting in error, and do not want to recover the settings you specified, then no action is required. If you want to specify the setting, make sure the setting is legal in the specified file and correct any syntax or semantic errors, then add the setting to the file again. You may need to recover a previous version of the file to determine the exact code the MAX+PLUS II Advanced Synthesis software excluded. Make sure the settings you add conform to the usage code statement definitions and the format specific to the settings and configuration file section. To avoid receiving this message in the future, Altera recommends using the MAX+PLUS II Advanced Synthesis software dialog boxes and wizards to modify settings, rather than editing the settings and configuration files manually.

Settings and configuration file <*name*> already contains section <*name*> -- deleting duplicate section

---

CAUSE: You added the specified section to the specified settings and configuration file, but the specified section already exists in the file. The MAX+PLUS II Advanced Synthesis software rewrote the file to exclude the duplicate section.

ACTION: If you added the duplicate section in error, and do not want to recover the settings you specified in the duplicate section, then no action is required. If you want to specify the deleted settings in the file, add the settings to the section that the MAX+PLUS II Advanced Synthesis software did not delete from the file. You may need to recover a previous version of the file to determine the exact code the MAX+PLUS II Advanced Synthesis software excluded. Make sure the settings you add conform to the usage code statement definitions and the format specific to the settings and configuration file section. To avoid receiving this message in the future, Altera recommends using the MAX+PLUS II Advanced Synthesis software dialog boxes and wizards to modify settings, rather than editing the settings and configuration files manually.

Settings and configuration file <*name*> already contains section <*name*>(<*name*>) -- deleting duplicate section

---

CAUSE: You added the specified section to the specified settings and configuration file, but the specified section already exists in the file. The MAX+PLUS II Advanced Synthesis software rewrote the file to exclude the duplicate section.

ACTION: If you added the duplicate section in error, and do not want to recover the settings you specified in the duplicate section, then no action is required. If you want to specify the deleted settings in the file, add the settings to the section that the MAX+PLUS II Advanced Synthesis software did not delete from the file. You may need to recover a previous version of the file to determine the exact code the MAX+PLUS II Advanced Synthesis software excluded. Make sure the settings you add conform to the usage code statement definitions and the format specific to the settings and configuration file section. To avoid receiving this message in the future, Altera recommends using the MAX+PLUS II Advanced Synthesis software dialog boxes and wizards to modify settings, rather than editing the settings and configuration files manually.

Settings and configuration file <*name*> contains a syntax error that causes an illegal setting -- deleting illegal setting

---

CAUSE: You edited the specified settings and configuration file manually, and you added a setting to the file, but the MAX+PLUS II Advanced Synthesis software detects a syntax error in the setting. The MAX+PLUS II Advanced Synthesis software rewrote the file to exclude any settings with syntax errors.

ACTION: If you added the illegal setting in error, and do not want to recover the setting, then no action is required. If you want to specify the deleted setting in the file, make sure the setting is legal for the specified file and correct any syntax or semantic errors, then add the setting to the file again. You may need to recover a previous version of the file to determine the exact code the MAX+PLUS II Advanced Synthesis software excluded. Make sure the settings you add conform to the usage code statement definitions and the format specific to the settings and configuration file section. To avoid receiving this message in the future, Altera recommends using the MAX+PLUS II Advanced Synthesis software dialog boxes and wizards to modify settings, rather than editing the settings and configuration files manually.

Settings and configuration file <*name*> contains a syntax error that causes an illegal setting. The incorrect text is <*text*> -- deleting illegal setting.

CAUSE: You edited the specified settings and configuration file manually, and you added a setting to the file, but the MAX+PLUS II Advanced Synthesis software detects a syntax error in the specified text. The MAX+PLUS II Advanced Synthesis software rewrote the file to exclude any settings with syntax errors.

ACTION: If you added the illegal setting in error, and do not want to recover the setting, then no action is required. If you want to specify the deleted setting in the file, make sure the setting is legal for the specified file and correct any syntax or semantic errors, then add the setting to the file again. You may need to recover a previous version of the file to determine the exact code the MAX+PLUS II Advanced Synthesis software excluded. Make sure the settings you add conform to the usage code statement definitions and the format specific to the settings and configuration file section. To avoid receiving this message in the future, Altera recommends using the MAX+PLUS II Advanced Synthesis software dialog boxes and wizards to modify settings, rather than editing the settings and configuration files manually.

Shift operator shift amount too wide : shift distance is <*number*> bits wide, data width is <*number*> bits

---

CAUSE: In a Verilog HDL or VHDL design file, you created a shift operator with the specified variable shift distance, which you specify using a register. However, the shift distance is greater than the specified width of the data that you are shifting. For example, this error can occur if you shift an 8-bit register by the value of a register with 4 or more bits, where the shift distance can be up to 16 bits.

This limitation will be removed in a future version of the MAX+PLUS II Advanced Synthesis software.

ACTION: Change the size of the shift distance register so the shift distance can never be greater than the data width.

See also:

Section 4.1.12 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language manual*
Section 7.2.3 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

Some WYSIWYG primitives POWER_UP_HIGH states changed

CAUSE: The `POWER_UP_HIGH` states of some WYSIWYG primitives have changed due to disconnected control signals, such as preset or clear. Control signals must be connected to maintain the `POWER_UP_HIGH` state.

ACTION: Verify that the logic feeding the control signals in the WYSIWYG primitive is correct, especially the logic feeding the preset and clear ports.

Starting second synthesis pass using Fitter timing information

---

CAUSE: You attempted to optimize netlists by directing the MAX+PLUS II Advanced Synthesis software to use timing information from the Fitter to improve the quality of synthesis. As a result, the MAX+PLUS II Advanced Synthesis software is running a second synthesis pass that uses timing information from the Fitter.

ACTION: If you want to use Fitter timing information in synthesis, no action is required. Otherwise, specify that the MAX+PLUS II Advanced Synthesis software not use timing information from the Fitter to improve the quality of synthesis.

State <*name*> uses code string <*text*>

CAUSE: The Logic Synthesizer is reporting the code string for the specified state in a state machine.

ACTION: No action is required.

State bit assignments are not unique for state <*name*> and state <*name*>

CAUSE: You have manually assigned the same state bit pattern to the two specified states in a state machine, or assigned some but not all state bits for the states in the machine. However, each state bit assignment in a state machine must be unique.

ACTION: No action is required. The Logic Synthesizer adds extra state bits to make each state bit assignment unique.

State machine <*name*> contains <*number*> states and <*number*> state bits

---

CAUSE: The Logic Synthesizer started processing the specified state machine, and is reporting the total number of states and state bits found in the state machine during preprocessing.

ACTION: No action is required.

State Machine <*name*> has no clock signal

CAUSE: In the Logic Section in a Text Design File (**.tdf**), you did not specify the clock of the specified state machine.

ACTION: Edit the design file and specify the clock signal of the state machine.

State machine <*name*> must be reset to ensure proper operation

---

CAUSE: The initial state in your state machine contains non-zero state bits. To ensure proper operation of the state machine, you must must explicitly reset the state machine so that the logic commences at the Reset state. This condition may occur as a result of logic synthesis if you turn on the **Not Gate Push-Back** option to invert a state bit that feeds an output pin.

ACTION: Reset the state bits to an initial condition in the design.

Support for IP core <*name*> in device family <*family*> is advanced and core functionality has not been fully verified

---

CAUSE: You instantiated the specified core in the specified device family and attempted to compile the design. However, support for the core is advanced for the device family, and the core functionality has not been fully verified. Altera recommends that you do not use this core in the selected device family for production designs.

ACTION: Contact the core vendor for more information.

Support for IP core <*name*> in device family <*family*> is preliminary and specifications are subject to change

---

CAUSE: You instantiated the specified core in the specified device family and attempted to compile the design. However, support for the core is preliminary for the device family and specifications are subject to change.

ACTION: No action is required. Refer to the core documentation for more information.

Technology mapper logic option must be set to LUT for FLEX 6000 device family

---

CAUSE: You did not specify the setting for the **LUT Technology Mapper** logic option, but Flex 6000 devices can be used only with the LUT **Technology Mapper** logic option. As a result, the MAX+PLUS II Advanced Synthesis software will ignore your assignment and change the **Technology Mapper** logic option value to **Look-up Table (LUT)**.

ACTION: No action is required.

Top-level design file project must contain output or bidirectional pins

---

CAUSE: The top-level design file of the current project does not contain any output pins or any bidirectional pins that are used as output pins. You must include at least one output or bidirectional pin in the top-level design file.

ACTION: Add at least one output pin or bidirectional pin used as an output pin to the top-level design file.

Top-level entity <*name*> is not from a Verilog or VHDL source.

CAUSE: The top-level entity did not come from either a Verilog or VHDL design source. Non-Verilog/VHDL top-level entities are not supported.

ACTION: Check the design to see if this is what the user intended.

TRI or OPNDRN buffer <*name*> already drives OUTPUT pin -- it cannot also drive other types of primitives

---

CAUSE: The specified `TRI` or `OPNDRN` buffer primitive drives an `OUTPUT` (or `OUTPUTC`) pin as well as other types of primitives. As a result, the MAX+PLUS II Advanced Synthesis software cannot compile the current design.

ACTION: Connect the `TRI` or `OPNDRN` pin to either an `OUTPUT` (or `OUTPUTC`) pin or a `BIDIR` (or `BIDIRC`) pin under the following conditions:

- If you use the `TRI` buffer to drive an `OUTPUT` pin, make sure the `TRI` buffer drives only other `OUTPUT` pins.
- If you use the `TRI` buffer to drive a `BIDIR` pin, make sure the `TRI` buffer does not drive other `OUTPUT` or `BIDIR` pins directly (that is, the `TRI` must be separated from the pins by logic).

TRI or OPNDRN buffers permanently disabled

CAUSE: You connected the output enable input signal of a `TRI` or `OPNDRN` buffer to `GND`. The tri-state buffer does not reduce to a simpler form. This condition may be the result of logic synthesis. If you connected the `TRI` or `OPNDRN` buffer connected to a bidirectional pin, the message indicates that the pin will always drive in.

ACTION: Check the design file for errors. Connect the input to a node other than `GND` if you want to enable the output enable input. You might also want to substitute an `INPUT` pin for the `BIDIR` pin (and remove the `TRI` or `OPNDRN` buffer) if the pin is always driving in.

TRI or OPNDRN buffers permanently enabled

CAUSE: You connected the output enable input of a `TRI` or `OPNDRN` buffer to `VCC` or a signal that reduces to `VCC`. As a result, the `TRI` or `OPNDRN` buffers are permanently enabled.

ACTION: If you want the buffer to be permanently enabled, no action is required. If you do not want the buffer to be permanently enabled, connect the input to a node other than `VCC`.

Tri-state bus <*name*> fed by inverted tri-state bus <*name*> but can be fed only by tri-state primitives

---

CAUSE: You assigned an inverted tri-state bus to a tri-state bus, but tri-state buses can only be fed by tri-state primitives.

ACTION: Replace the inverted tri-state bus with a tri-state primitive (such as `TRI` or `OPNDRN`).

Tri-state bus fed by inverted TRI primitive <*name*> but can be fed only by tri-state primitives positively

---

CAUSE: You assigned the specified `TRI` primitve to a tri-state bus with an inversion between them, but tri-state buses can only be fed by tri-state primitives positively.

ACTION: Check the design and make sure the tri-state bus is being fed by tri-state primitives only (such as `TRI` or `OPNDRN`).

Tri-state bus fed by primitive <*type*> <*name*>, but can be fed only by tri-state primitive

---

CAUSE: You fed a primitive that is not a `TRI` primitive to a tri-state bus, but tri-state buses can only be fed by tri-state primitives.

ACTION: Two primitives are driving the same wire, causing the line to become a tri-state bus. If this is what you intended, add a tri-state primitive (such as `TRI` or `OPNDRN`) between the bus and the other primitive. If you did not intend to create a tri-state bus, correct the netlist so that only one primitive drives the wire.

Two or more logical RAMs use the same name

CAUSE: Two or more logical RAMs in the design use the same name. However, all logical RAMs must have unique names.

ACTION: Modify the design so that all logical RAMs have unique names.

Unsupported Verilog HDL feature at <*location*>: variable index to multidimensional array is not supported

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used a variable index to a multidimensional array. Variable indexes to multidimensional arrays are not supported by the MAX+PLUS II Advanced Synthesis software.

ACTION: Edit the design to change the multidimensional array into multiple stages of one-dimensional arrays.

See also:

Section 3.10 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Unsupported Verilog HDL feature error at <*location*>: real number in conditional operator (?:) expression is not supported

---

CAUSE:	In a Verilog Design File (**.v**) at the specified location, you used a real number as the condition to check in a conditional operator (`?:`) expression. Although Verilog HDL supports real numbers in conditional operator expressions, the MAX+PLUS II Advanced Synthesis software does not support the use of real numbers in conditional operator expressions.
ACTION:	Make sure the conditional operator expression does not contain real numbers.

Using Fitter timing information for synthesis optimization

CAUSE: You attempted to optimize netlists by directing the MAX+PLUS II Advanced Synthesis software to use timing information from the Fitter to improve the quality of synthesis. As a result, the MAX+PLUS II Advanced Synthesis software is running a second synthesis pass that uses timing information from the Fitter.

ACTION: If you want to use Fitter timing information during synthesis, no action is required. Otherwise, specify that the MAX+PLUS II Advanced Synthesis software not use timing information from the Fitter to optimize netlists during synthesis.

Value <*text*> for LPM_HINT parameter contains syntax error(s)

CAUSE: The value you have specified in the source file for the `LPM_HINT` parameter contains syntax error(s). If the parameter contains one or more subparameters, the syntax should be as follows:

```
sub_param1 = value1 [, sub_param2 = value2]
```

ACTION: Make sure the `LPM_HINT` parameter uses the correct syntax.

Value <*value*> for <*name*> parameter in WYSIWYG primitive <*name*> must be <*values*>.

CAUSE: The specified WYSIWYG primitive has an illegal value for the specified parameter. The message text specifies possible legal values.

ACTION: Modify the design to specify a legal parameter value for the WYSIWYG primitive. If the value is correct, and you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

Value for <*name*> parameter in WYSIWYG primitive <*name*> must be greater than or equal to <*number*>

CAUSE: The specified WYSIWYG primitive has the specified parameter, but the value for the parameter is less than the minimum number allowed. The value must be greater than or equal to the specified minimum number.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

Value for <*name*> parameter in WYSIWYG primitive <*name*> must be less than or equal to <*number*>

CAUSE: The specified WYSIWYG primitive has the specified parameter, but the value for the parameter is greater than the maximum number allowed. The value must be less than or equal to the specified minimum number.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

Value for <*name*> parameter in WYSIWYG primitive <*name*> must be ON or OFF

CAUSE: The specified WYSIWYG primitive has the specified parameter, but the parameter has a value that is not `ON` or `OFF`. The value of the specified parameter must be `ON` or `OFF`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

Value for <*name*> parameter in WYSIWYG primitive <*name*> must be TRUE or FALSE

CAUSE: The specified WYSIWYG primitive has the specified parameter, but the parameter has a value that is not TRUE or FALSE. The value of the specified parameter must be TRUE or FALSE.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

Value for REGISTER_CASCADE_MODE parameter in WYSIWYG primitive <*name*> must be ON or OFF

CAUSE: The specified WYSIWYG primitive has an illegal value for the `REGISTER_CASCADE_MODE` parameter. The `REGISTER_CASCADE_MODE` parameter must be set to `ON` or `OFF`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

Value for SUM_LUTC_INPUT parameter in WYSIWYG primitive <*name*> must be DATAC, CIN, or QFBK

CAUSE: The specified WYSIWYG primitive has an illegal value for the `SUM_LUTC_INPUT` parameter. The `SUM_LUTC_INPUT` parameter must be set to `DATAC`, `CIN`, or `QFBK`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

Value for SYNCH_MODE parameter in WYSIWYG primitive <*name*> must be ON or OFF

CAUSE: The specified WYSIWYG primitive has an illegal value for the `SYNCH_MODE` parameter. The `SYNCH_MODE` parameter must be set to `ON` or `OFF`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

Verilog HDL Always Construct error at <*location*>: Force Statement is not supported for processing with Integrated Synthesis

---

CAUSE:	In an Always Construct at the specified location in a Verilog Design File (**.v**), you used a Force Statement. Although Force Statements are supported in Verilog HDL, they are not supported for processing with Integrated Synthesis. A Force Statement is used in conjuction with a Release Statement to override values on wires or registers. It is typically used with a simulator during design debugging to change signal values; however, it is not supported for synthesis, because you should never need to release the value on a wire or in a net in a synthesized design.
ACTION:	Edit the design to remove the Force Statement.

See also:

Sections 5.6.2 and 9.3 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Always Construct error at <*location*>: Forever Statement is not supported in Always Construct

---

CAUSE: In an Always Construct at the specified location in a Verilog Design File (**.v**), you used a Forever Statement. However, although Forever Statements are supported in both Always Constructs and Initial Constructs in Verilog HDL, the MAX+PLUS II Advanced Synthesis software supports Forever Statements only in Initial Constructs.

ACTION: Move the Forever Statement from the Always Construct to an Initial Construct, or replace the Forever Statement in the Always Construct with one of the other types of Looping Statements.

See also:

Section 9.6 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Case Statement error at <*location*>: generated case expression is not constant

---

CAUSE: In a Case Statement at the specified location in a Verilog Design File (**.v**), you used a generated case expression that is not a constant. The

ACTION: Edit the case expression to be a compile-time constant function.

See also:

Section 9.5 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Case Statement error at <*location*>: ignoring Case Statements that contain X or Z value

---

CAUSE: In a Case Statement at the specified location in a Verilog Design File (**.v**), you used a don't care (X) or high impedance (Z) value. Case Statements that contain X or Z values are not supported by Integrated Synthesis and will be ignored.

ACTION: Edit the design to specify numeric values for Case Statements.

See also:

Section 9.5 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Case Statement error at <*location*>: must use no more than one Default Statement

---

CAUSE: In a Case Statement at the specified location in a Verilog Design File (**.v**), you have specified more than one Default Statement; however, you must specify no more than one Default Statement in a Case Statement.

ACTION: Remove the extra Default Statements from the Case Statement until it contains no more than one Default Statement.

See also:

Section 9.5 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Case Statement error: generated case item expression at <*location*> is not constant

---

CAUSE: In a Case Statement at the specified location in a Verilog Design File (**.v**), you used a generated case item expression that is not a constant. Case Statements must use constant values.

ACTION: Rewrite the case item expression of the Case Statement to be a compile-time constant function.

See also:

Section 9.5 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Case Statement warning at <*location*>: case item expression is ignored because it never applies

---

CAUSE: In a Case Statement at the specified location in a Verilog Design File (**.v**), you specified a case item expression that has more significant bits than the number of bits in the case expression that it is being compared against. As a result, the case item expression can never apply, and is ignored by the MAX+PLUS II Advanced Synthesis software.

ACTION: No action is required. To avoid receiving this message in the future, remove bits from the case item expression until the bit length matches that of the case expression.

See also:

Section 9.5 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Case Statement warning: implemented Verilog HDL full_case directive at <*location*> -- differences between design synthesis and simulation may occur

---

CAUSE: At the specified location in a Verilog Design File (**.v**), you used the `full_case` directive in a Case Statement that does not contain all possible state values or a Default Statement. The `full_case` directive directs the Logic Synthesizer to treat unspecified state values in the Case Statement as don't care values. The Logic Synthesizer implemented the `full_case` directive; however, in doing so, the Logic Synthesizer may have created synthesized logic for the current design with functionality that differs from the functionality that you simulated for the design.

ACTION: If possible, change the Case Statement so it is "full," that is, all the state values in the Case Statement are specified or the Case Statement contains a Default Statement. You can then remove the `full_case` directive from the Case Statement.

See also:

full_case language directive
Verilog HDL Language Directives

Verilog HDL Case Statement warning: implemented Verilog HDL parallel_case directive at <*location*> -- differences between design synthesis and simulation may occur

---

CAUSE:	At the specified location in a Verilog Design File (**.v**), you used the `parallel_case` directive in a Case Statement that does not contain mutually exclusive case item expressions. The `parallel_case` directive directs the Logic Synthesizer to implement parallel logic rather than a priority scheme for all case item expressions in the Case Statement. The Logic Synthesizer implemented the `parallel_case` directive; however, in doing so, the Logic Synthesizer may have created synthesized logic for the current design with functionality that differs from the functionality you simulated for the design.
ACTION:	Check the synthesized logic for design integrity.

See also:

parallel_case language directive
Verilog HDL Language Directives

Verilog HDL Compiler Directive error at <*location*>: can't open Verilog Design File <*name*>

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used an `` `include `` compiler directive. However, the MAX+PLUS II Advanced Synthesis software cannot open the specified Verilog Design File. This error may occur if both the design file and the included file are located in a lower-level directory instead of at the project root directory, if the included file is an encrypted megafunction design file for which you do not have a license, or if did not specify the correct file name or path name of the included Verilog Design File in the Include Statement.

ACTION: Make sure that all design files are in the correct locations, that any megafunctions you reference are correctly licensed, and that you have spelled the file names and path names of included files correctly. Make sure all included files are either in the project root directory, or are referenced by their full path from the project root directory.

See also:

Sections 13.2 and 19.5 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Compiler Directive error at <*location*>: incorrect number of text macro arguments

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used a text macro name with a different number of arguments than the arguments that appear in the Text Macro Definition.

ACTION: Edit the design to make sure the number of arguments in the call to the text macro match the number of arguments in the Text Macro Definition.

See also:

Section 19.3 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Compiler Directive error at <*location*>: incorrect use of predefined text macro <*name*> -- expected macro field <*text*>

CAUSE:	In a Verilog Design File (**.v**) at the specified location, you used a recognized Compiler Directive (such as `` `define ``); however, the remaining syntax for the Compiler Directive is incorrect. As a result, Integrated Synthesis could not parse the specified macro field.
ACTION:	Edit the design to make sure the correct syntax is used in the Compiler Directive and in the text macro field.

See also:

Section 19.3 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Compiler Directive error at <*location*>: missing arguments for text macro

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used a text macro name that has no arguments; however, the Text Macro Definition does have arguments. The number of arguments in the call to the macro must match the number of arguments in the Text Macro Definition.

ACTION: Edit the design to make sure the number of arguments in the macro name match the number of arguments in the Text Macro Definition.

See also:

Section 19.3 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Compiler Directive error at <*location*>: missing Compiler Directive

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used what appears to be a Compiler Directive (such as `` `define ``); however, the Compiler Directive keyword you used is not recognized by Integrated Synthesis.

ACTION: Edit the design to use the correct Compiler Directive keyword.

See also:

Section 19 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Compiler Directive error at <*location*>: must use `ifdef directive with `else directive

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used an `else directive, but did not specify a corresponding `ifdef directive.

ACTION: Add an `ifdef directive or remove the `else directive.

See also:

Section 19.4 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Compiler Directive error at <*location*>: must use `ifdef directive with `elsif directive

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used an `elsif directive, but did not specify a corresponding `ifdef directive.

ACTION: Add an `ifdef directive or remove the `elseif directive.

See also:

Section 19.4 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Compiler Directive error at <*location*>: must use `ifdef directive with `endif directive

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used an `endif directive, but did not specify a corresponding `ifdef directive.

ACTION: Add an `ifdef directive or remove the `endif directive.

See also:

Section 19.4 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Compiler Directive error at <*location*>: predefined text macro <*name*> cannot be undefined

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used an `` `undef `` Compiler Directive for the specified macro name; however, the specified macro is a predefined macro and cannot be undefined.

ACTION: Remove the `` `undef `` directive.

See also:

Section 19.3 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Compiler Directive error at <*location*>: text macro <*name*> is same as predefined text macro

---

CAUSE: In a Text Macro Defintion at the specified location in a Verilog Design File (**.v**), you defined a text macro name that is the same name as a predefined text macro.

ACTION: Define the text macro using a different name.

See also:

Section 19.3 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Compiler Directive error at <*location*>: text macro <*name*> is undefined

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you attempted to use the specified macro by typing the (`) character followed by the text macro name, but the macro has not been defined with a `` `define `` directive. This message may occur if you forgot to define the macro, or it may also occur if you were not intending to use a macro, but accidentally typed an accent grave or "back tick" character (`).

ACTION: Define the macro, or if you were not intending to use a macro, remove the ` character.

See also:

Section 19 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Compiler Directive error at <*location*>: too many arguments in text macro

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you created or used a text macro with more than 128 arguments. The MAX+PLUS II Advanced Synthesis software requires that a text macro have no more than 128 arguments.

ACTION: Reduce the number of arguments for the macro so it does not exceed 128.

See also:

Section 19 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Compiler Directive warning at <*location*>: illegal value for directive unconnected_drive <*name*> -- value is ignored

---

CAUSE:	In a Verilog Design File (**.v**) at the specified location, you used the `` `unconnected_drive `` Compiler Directive with the specified value; however, the value you specified is not legal for the `` `unconnected_drive `` directive. The only legal values are `pull0` and `pull1`.
ACTION:	No action is required. To avoid receiving this message in the future, specify a value of `pull0` or `pull1` for the `` `unconnected_drive `` directive.

See also:

Section 19.9 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Conditional Statement error at <*location*>: generated if condition is not constant

---

CAUSE: In a Conditional Statement at the specified location in a Verilog Design File (**.v**), the generated if condition expression is not a constant.

ACTION: Rewrite the if expression of the Conditional Statement to be a compile-time constant function.

See also:

Section 9.5 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Conditional Statement error at <*location*>: If-Else Statement does not match any sensitivity list edge

---

CAUSE: In a Conditional Statement at the specified location in a Verilog Design File (**.v**), you specified an If-Else Statement inside an Always Construct that does not match any edge on the sensitivity list of the Always Construct. This error may occur if you are trying to model a `DFF` with multiple control signals. The MAX+PLUS II Advanced Synthesis software will only infer a secondary signal from a single secondary signal in an if condition. For example, you may have written the following sample structure to model a `DFF` primitive that can be reset by two signals, `rst1` or `rst2`:

```
always @ (posedge clk or posedge rst1 or posedge rst2)
begin
  if (rst1 == 1'b1 || rst2 == 1'b1)
    q <= 1'b0;
  else
    q <= d;
end
```

The MAX+PLUS II Advanced Synthesis software will give an error for this sample Conditional Statement because it contains an expression with two sensitivity list edges.

ACTION: Edit the design to specify only one edge per if condition. For example, if you were to edit the previous example to specify only one edge per if condition, the MAX+PLUS II Advanced Synthesis software would then succesfully recognize the `DFF` primitive. The sample code would then appear as follows:

```
always @ (posedge clk or posedge rst1 or posedge rst2)
begin
  if (rst1 == 1'b1)
    q <= 1'b0;
  else if (rst2 == 1'b1)
    q <= 1'b0;
  else
    q <= d;
end
```

Alternatively, you could generate the `OR` of `rst1` and `rst2` outside the Always Construct.

See also:

Section 9.4 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Defparam Statement error at <*location*>: value for parameter <*name*> must be constant expression

---

CAUSE: In a Defparam Statement at the specified location in a Verilog Design File (**.v**), you specified a value for the specified parameter that is not a constant expression. You must specify only constant expressions for parameter values.

ACTION: Change the parameter value to a constant expression.

See also:

Section 12.2.1 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: array <*name*> should be indexed by <*number*> dimensions

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you specified the wrong number of dimensions while accessing a multidimensional array. When accessing a multidimensional array, you should normally specify exactly one index for every dimension. However, the MAX+PLUS II Advanced Synthesis software does not currently support synthesis of multidimensional arrays. If you fix this error, you will still receive an unsupported feature error for using a multidimensional array.

ACTION: Edit the design to remove the use of multidimensional arrays. As an alternative, you may want to use a memory, which is an array of vectors and can be declared as shown in the following example:

```
reg [n:0] my_memory[m:0]
```

See also:

Section 3.10 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: assignment to illegal expression

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you made an assignment to an illegal expression, or an expression that is not supported by the MAX+PLUS II Advanced Synthesis software.

ACTION: Make sure the assigned expression is a legal expression.

See also:

Section 6 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: can't find port <*name*>

CAUSE: In a Verilog Design File (**.v**) at the specified location, you referenced the specified port, but that port name does not exist in the list of ports for the Module Definition.

ACTION: Make sure the port name reference matches the port name specified in the list of ports for the Module Definition.

See also:

Section 12.1 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: constant index of multidimensional array is not within range

---

CAUSE:	In a Verilog Design File (**.v**) at the specified location, you used a constant index to a multidimensional array, but the index is not within the specified range.
ACTION:	Verify the correctness of the constant index or increase the array size.

See also:

Sections 3.10 and 7.1 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: function <*name*> is used but is not declared

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used the specified function, but the function is not declared.

ACTION: Edit the design to use a Function Declaration in the module in which the function is used.

See also:

Section 10.3 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: generated variable <*name*> cannot have assignment

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you attempted to make an assignment to a generated variable, but you cannot make assignments to generated variables.

ACTION: Remove the assignment to the generated variable.

See also:

Section 12.1 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: hierarchical name <*name*> cannot reference signal in another hierarchy

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used a hierarchical name to reference a signal in another hierarchy. The MAX+PLUS II Advanced Synthesis software does not allow you to reference an internal signal of another module.

ACTION: Change the design to reference only signals that are declared within the current module, or its inputs.

See also:

Section 12 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: identifier <*name*> cannot be used in expression

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used an identifier in an expression, but the identifier does does not evaluate to a data type. For example, the identifier may be a module name.

ACTION: Remove the specified identifier from the expression.

See also:

Section 2.7 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: identifier <*name*> is not a memory

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used the specified identifier, but the MAX+PLUS II Advanced Synthesis software expected this identifier to be declared as a memory.

ACTION: If you intended this identifier to be a memory, make sure it is declared as a memory. If you did not intend this identifier to be a memory, edit the design to make sure it does not refer to this identifier as a memory.

See also:

Section 3.10 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: illegal binary operation on integers

---

CAUSE:	In a Verilog Design File (**.v**) at the specified location, you used a binary operator with an integer variable; however, the operator you used is not supported for integers.
ACTION:	Edit the design to use a different variable type or to use a different operator.

See also:

Section 4.1 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: illegal binary operation on real numbers

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used a binary operator in a real number expression; however, the operator you used is not supported for real number expressions. Although, Integrated Synthesis supports arithmetic functions in real number expressions, it does not support bit-wise and logical operations in real number experessions.

ACTION: Edit the design to use an integer or reg variable type if you require bit-wise operations.

See also:

Section 4.1 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: illegal bit-wise unary operation on real number

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used a bit-wise unary operator in a real number expression; however, bit-wise operators are not supported for real number expressions.

ACTION: Edit the design to use an integer or reg variable type if you require bit-wise operations.

See also:

Section 4.1 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: illegal name <*name*> used in expression

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used a name in an expression, but the name is not legal for use in the expression. For example, this error may occur if you instantiate a module, and then try to assign the instantiated module to another signal, as shown in the following example:

```
sub sub_inst (in1, in2, temp);
assign out = sub_inst;
```

ACTION: Remove the specified name from the expression.

See also:

Section 4 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: illegal operation on real number

---

CAUSE:	In a Verilog Design File (**.v**) at the specified location, you used a bit-wise operation in a real number expression; however, real number expressions do not support bit-wise operations.
ACTION:	Edit the design to use an integer or reg variable type if you require bit-wise operations.

See also:

Section 4.1 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: index of variable <*name*> is not within address range

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you assigned a value to the specified variable at a bit index that is not within the declared range for the address. The Compiler is ignoring the variable assignment.

ACTION: Edit the design to make sure the index of the variable is within the declared range.

See also:

Section 3.2 and 6.2 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: indexed object <*name*> cannot have assigned value

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you assigned a value to the specified indexed object. However, value must not be assigned to an indexed object of that type.

ACTION: Remove the assignment to the indexed object.

See also:

The IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: left index of part-select of variable <*name*> is out of address bounds

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used a part-select, but the left index on the part-select is greater than the declared address range for the specified variable.

ACTION: Edit the part-select index so it is within the declared address range for the variable.

See also:

Section 4 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: left index of part-select of variable <*name*> is out of address bounds -- returning don't care (X) value

---

CAUSE:	In a Verilog Design File (**.v**) at the specified location, you used a part-select for the specified variable. However, the left index of this part-select is greater than the declared address range for the variable. The return value of the expression is being considered as don't care (X).
ACTION:	Edit the design to make sure the part-select is within the declared address range for the variable.

See also:

Section 4 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: memory cannot be accessed directly

---

CAUSE: In an Always Construct, at the specified location in a Verilog Design File (**.v**), you declared an array for direct memory access; however, memory must not be accessed directly.

ACTION: Specify access to the memory explicitly, bit by bit, as shown in the following example:

```
module mem_fixed(x);
output [1:0] x;
reg mem1bit[1:0];
always
begin
    x[1] = mem1bit[1];
    x[0] = mem1bit[0];
end
endmodule
```

See also:

Section 3.10 and 4.2 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: memory size reaches 2**<*number*> bits limit

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you specified a memory size; however, the memory size you specified is greater than or equal to the limit of 2**28 bits.

ACTION: Reduce the memory size to less than 2**28 bits.

See also:

Section 3.10.3 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: must use only constant operands for operator

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you have used a non-constant operand, such as a `reg`, with an operator that requires only constant operands in the MAX+PLUS II Advanced Synthesis software.

ACTION: Modify the design to use a different operator, or change the operand to a constant value.

Verilog HDL error at <*location*>: name <*name*> is not a task or block

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used the specified name to refer to another block or task, but the name is not defined as a block or task.

ACTION: Declare the task or block with a Task Declaration or Specify Block Declaration, or make sure the design uses the correct name when refering to a task or block.

See also:

Sections 9.8 and 10 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: object <*name*> cannot have assigned value

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you assigned a value to the specified object; however, you must not assign value to an object of that type.

ACTION: Remove the assignment to the object.

See also:

Section 6 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: parameter <*name*> is not defined in module

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used a Defparam Statement or a named Module Instance Parameter Value Assignment list to change the value of a parameter in a module that you are instantiating. However, the specified parameter does not exist in the instantiated module. This error may occur if you have mistyped the parameter name.

ACTION: Carefully check the names in the Parameter Value Assignment Statement and in the instantiated module to make sure they match.

See also:

Section 12 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: part-select direction is opposite from prefix index direction

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used a part-select to select a part of a vector; however, in the part-select, the direction from MSB to LSB is reversed from the direction in the declaration of the vector. For example, see the following excerpt of a sample design, which uses a part-select on vector:

```
module veri_test(in, out);
        input [3:0] in;
        output [1:0] out;
        assign out = in[0:1];
        endmodule
```

ACTION: Use the same direction in the part-select of a vector as is used in the declaration of the vector. For example, in the previous sample design, you can change the assignment to `out` into the following format:

```
assign out = in[1:0];
```

See also:

Section 4 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: part-select has negative size, but must use zero or more bits

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you have used a part-select of a vector with a negative size. However, the part-select must select at least zero or more bits.

ACTION: Make sure all part-selects in the design select a partial array of zero or more bits of their vector.

See also:

Section 3.10 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: part-select of memory <*name*> is not allowed

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you have tried to use a part-select of the specified memory. However, you must select only a single complete memory word using a part-select. You cannot access several words at a time using a part-select, nor can you access bits within a single memory word using a part-select.

ACTION: If you want to access several memory words, rewrite your design to use a For loop to access each word. If you want to select part of one memory word, select the complete memory word, assign it to a temporary register, and then use a part-select on the temporary register to access the bits you want.

See also:

Section 3.10 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: ports are defined with expressions -- must use standard Verilog HDL statements to instantiate modules

---

CAUSE:	In a Verilog Design File (**.v**) at the specified location, you defined some ports by using Verilog HDL expressions. You must use standard Verilog HDL statements to instantiate modules.
ACTION:	Use standard Verilog HDL statements, such as Module Definition and Module Instantiation statements, to instantiate modules.

See also:

Section 12.1 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: ports in concatenation have different directions

---

CAUSE: In a Module Declaration at the specified location in a Verilog Design File (**.v**), you specified a port expression using a concatenation, but some of the ports in the concatenation have different directions.

ACTION: Make sure all ports in the concatenation have the same direction.

See also:

Sections 4.1.14, 4.2.3, and 12.3.2 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: range index cannot be a real number

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used a floating-point value when defining a range. Since a range represents a number of bits, a real number value is not legal in a range.

ACTION: Make sure all values in the range specification evaluate to an integer.

See also:

Sections 3.9 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: range index cannot contain X or Z

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used a don't care (X) or high impedance (Z) value in defining a range. However, a range represents a number of bits, so an X or Z value is not legal for a range specification.

ACTION: Edit the design so all values in range specifications evaluate to a constant integer.

See also:

Sections 3.3 and 7.1 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: range index is not constant

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used a nonconstant value in defining a range. A range must not have a variable size.

ACTION: Edit the design so all values in range specifications evaluate to a constant integer.

See also:

Sections 3.3 and 7.1 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: replication multiplier contains X

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used a replication operation with a multiplier value that contains a don't care (X) value. A replication operation must contain a non-zero, non-X, and non-Z constant expression.

ACTION: Make sure the multiplier value in a replication operation is a non-zero, non-X, and non-Z constant expression.

See also:

Section 4.1.14 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: replication multiplier is not constant

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used a replication operation with a multiplier value that is not constant, for example, `assign out = {myvar{in}}` where `myvar` is a reg or integer. The multiplier value in a replication operation must be a constant so that the width of the expression can be computed during synthesis.

ACTION: Edit the design to make sure all replication statements have constant multipliers. For example, in the following excerpt of a sample design, the `i` variable in the replication operation would be reported as an error, even though the design specifies that the value of `i` will always be `4`. To correct this error, you should replace the variable `i` in the replication operation with the constant value `4`.

```
reg [1:0] in;
integer i = 4;
assign out = {i{in};
```

See also:

Section 4.1.14 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: replication multiplier must be positive

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used a replication operation with a multiplier value that is negative.

ACTION: Edit the design to make sure the multiplier value in the replication operation is positive.

See also:

Section 4.1.14 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: right index of part-select of vector <*name*> is not within address range

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used a part-select, but the right index of the part-select is not within the address range of the vector.

ACTION: Change the right index of the part-select so it is within the address range of the vector.

See also:

Section 4.2.1 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: right index of part-select of vector <*name*> is not within address range

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used a part-select, but the right index of the part-select is not within the address range of the vector.

ACTION: Change the right index of the part-select so it is within the address range of the vector.

See also:

Section 4.2.1 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: sliced object <*name*> cannot have assigned value

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you assigned a value the specified sliced object. However, you must not assign a value to a sliced object of that type.

ACTION: Remove the assignment to the sliced object.

See also:

Section 6 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: system call <*name*> must have exactly one argument

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used the system call `$signed` or `$unsigned`. You either did not specify any arguments or specified two or more arguments. Both of the calls require exactly one argument.

ACTION: Make sure the system call has exactly one argument.

See also:

Section 17 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: task <*name*> is not declared

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used a task name that has not been declared. This message may occur if the task name does not match the name in the Task Declaration, or because the task was not declared correctly in a Task Declaration.

ACTION: Make sure the task name matches the task name in the Task Declaration, or declare the task in a Task Declaration.

See also:

Section 10 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: Task Enable Statement must not be used outside of sequential constructs

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used a Task Enable Statement in a concurrent construct. However, tasks can be called only from sequential constructs.

ACTION: Move the Task Enable Statement to a sequential construct.

See also:

Section 10.2 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: value cannot be assigned to constant

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you assigned a value to a constant wire; however, assignments to constants are not allowed. This error may occur if, for example, `w` is declared as `wire [1:0]` and you assign `w[2] = 1;`.

ACTION: Edit the design so the data type and bus width are correct and do not create an assignment to a constant.

See also:

Sections 2 and 6 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: value must not be assigned to nonvariable <*name*>

CAUSE: In a Verilog Design File (**.v**) at the specified location, you assigned a value directly to a nonvariable; however, a value must not be assigned to a nonvariable.

ACTION: Remove the assignment or create a new variable for the assignment.

See also:

Section 6.2 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: values cannot be assigned directly to memory <*name*>

CAUSE: In a Verilog Design File (**.v**) at the specified location, you assigned values directly to the specified memory, but you must not assign values to an entire memory directly.

ACTION: Assign values explictly to individual memory bits, as shown in the following example:

```
module mem_fixed(a, x);
input [1:0] a;
output x;
reg mem1bit[1:0];
always
begin
    mem1bit[1] = a[1];
    mem1bit[0] = a[0];
end
endmodule
```

See also:

Section 6.2 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: variable <*name*> cannot be indexed because it is not declared as an array

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you indexed the specified variable, but the variable cannot be indexed because it is not declared as an array.

ACTION: Declare the variable as an array with the proper bit width.

See also:

Section 3.10 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: variable <*name*> cannot be part-selected because it is not declared as an array

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used a part-select for the specified variable, but the variable cannot be part-selected because it is not declared as an array.

ACTION: Declare the variable as an array with the proper bit width.

See also:

Sections 3.10 and 4.2.1 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: variable <*name*> has mixed blocking and nonblocking Procedural Assignments -- must be all blocking or all nonblocking assignments

---

CAUSE:	In a Verilog Design File (**.v**) at the specified location, you made both blocking and nonblocking Procedural Assignments to one variable in the same Always Construct. Procedural Assignments to the same variable must either all be blocking or all be nonblocking.
ACTION:	Change or delete one or more assignments so that the Procedural Assignments for the variable are either all blocking or all nonblocking.

See also:

Section 6 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: variable <*name*> is not an array of vectors

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used too many indexes with a multidimensional array. For example, the following excerpt of a sample Verilog HDL design shows code that could cause this error:

```
reg mem[7:0][7:0];
always @(clk)
      begin
            out = mem[in_a][in_b][in_c];
            mem[in_a][in_b] = in_c;
      end
```

However, the MAX+PLUS II Advanced Synthesis software does not support synthesis of multidimensional arrays. If you fix this error, you will still receive an unsupported feature error for using a multidimensional array.

ACTION: Edit the design to remove the use of multidimensional arrays. As an alternative, you may want to use a memory, which is an array of vectors and can be declared as shown in the following example:

```
reg [n:0] my_memory[m:0]
```

See also:

Section 3.10 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: variable <*name*> is not declared

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used the specified variable, but the variable is not declared. This error can also occur if you reference a hierarchical name. Hierarchical names are not supported in MAX+PLUS II Verilog HDL, except in Defparam Statements. For more information on which Verilog HDL hierarchical structures are supported in the MAX+PLUS II Advanced Synthesis software, refer to Hierarchical Structures in the MAX+PLUS II Verilog HDL Support topic of MAX+PLUS II Help.

ACTION: Declare the variable, or make sure that you are not referencing a hierarchical name.

See also:

Section 6.2 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: variable <*name*> is not synthesizable because it does not hold its value under NOT (clock-edge) conditions

---

CAUSE:	In a Verilog Design File (**.v**) at the specified location, you used the specified variable, but the variable is not synthesizable because it does not hold its value under NOT (clock-edge) conditions.
ACTION:	Remove any code that assigns a value to the variable when the clock is not rising or falling.

See also:

Section 6.2 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error at <*location*>: variable index to array <*name*> is not within range

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used a variable index to the specified array, but the variable index is not within the boundaries of the range specified for the variable.

ACTION: Increase the array size to include the whole variable value range.

See also:

Sections 3.2, 3.10, and 7.1 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL error: can't create symbol for module <*name*> -- Port Declaration for port <*name*> cannot be a multidimensional array type

---

CAUSE:	You tried to create a symbol for the specified Verilog Design File (**.vhd**), but the Verilog Design File has a Port Declaration that is a multidimensional array type. The MAX+PLUS II Advanced Synthesis software cannot create a symbol for a module with a multidimensional array type.
ACTION:	Change the port types of the Verilog HDL module to be simple ports or one-dimensional array ports.

See also:

Creating a Block Symbol File for a Current File

Verilog HDL error: ignoring module <*name*> due to previous errors at <*location*>

CAUSE: In a Verilog Design File (**.v**) at the specified location, Integrated Synthesis cannot continue processing the specified module due to previously listed error messages.

ACTION: Correct the previous errors to allow module processing to continue.

Verilog HDL Event Control Statement error at <*location*>: Event Control Statement must be inside of Always Construct or Initial Construct

---

CAUSE:	In a Verilog Design File (**.v**) at the specified location, you used an Event Control Statement, but did not place it inside either an Always Construct or an Initial Construct. The Event Control Statement must always be inside either an Always Construct or an Initial Construct.
ACTION:	Move the Event Control Statement into an Always Construct or an Initial Construct.

See also:

Section 9.7 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Event Control Statement error at <*location*>: mixed single- and double-edge expressions are not supported

---

CAUSE: In an Event Control Statement at the specified location in a Verilog Design File (**.v**), you specified an Event Control Statement that contains both double-edge events and single-edge events. A double-edge event is a signal that is sensitive to both positive and negative edges, for example, `always @(A)`, and usually represents combinational logic. A single-edge event is a signal that is sensitive to just one edge, for example `always @(posedge B)`, and usually represents sequential logic (logic with clocked registers). You cannot include both event types in one Event Control Statement.

ACTION: Change the type of events in the Event Control Statement to be all single-edge or all double-edge events, or split the Event Control Statement into multiple statements.

See also:

Section 9.7 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Event Control Statement error at <*location*>: name <*name*> is not an event

---

CAUSE: In an Event Control Statement Verilog Design File (**.v**) at the specified location, you used the @ character to create an event. However, the specified name, which was used after the @ character, can not be an event.

ACTION: Replace the name with something that is an event.

See also:

Section 9.7 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL For Statement error at <*location*>: loop count exceeds limit

---

CAUSE: In a For Statement at the specified location in a Verilog Design File (**.v**), you specified a termination condition for the For loop that is never false, which causes an infinite loop during expansion.

ACTION: Make sure the terminating condition and assignment for the For loop are correct.

See also:

Section 9.6 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL For Statement error at <*location*>: must use only constant expressions in terminating conditions

---

CAUSE: In a For Statement at the specified location in a Verilog Design File (**.v**), you specified a terminating condition for a For loop that contains an arbitrary limit variable; however, only constant expressions are supported in For loop conditions.

ACTION: Make sure the terminating condition for the For loop contains only constant expressions.

See also:

Section 9.6 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Function Call error at <*location*>: incorrect number of arguments passed to function

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used a Function Call, but the Function Call does not pass the correct arguments to the function. The number of arguments passed must match the number of arguments in the Function Definition exactly.

ACTION: Edit the design to make sure the argument list in the Function Call matches the argument list in the Function Definition.

See also:

Sections 4 and 10.3 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Function Call or Function Declaration error at <*location*>: identifier <*name*> is not a function

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, either you used a Function Call using a name that is already declared as another nonfunction object, or, in a Function Declaration, you used a name that is already declared in your design as some other type of object.

ACTION: Make sure the Function Call name is spelled correctly, or change the name of the function so that it does not conflict with other identifiers.

See also:

Section 10.3 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Function Declaration error at <*location*>: Function Declaration cannot specify both range and type

---

CAUSE: In a Function Declaration at the specified location in a Verilog Design File (**.v**), you specified both a range and a type; however, you must not set both range and type for a Function Declaration.

ACTION: Remove either the range or the type specification in the Function Declaration.

See also:

Section 10.3 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Function Declaration error at <*location*>: must not use output or inout ports in Verilog HDL functions

---

CAUSE: In a Function Declaration at the specified location in a Verilog Design File (**.v**), you declared a function that has an `output` or `inout` port. However, Verilog HDL functions must have only `input` ports. You must not specify an `output` or `inout` port for a Verilog HDL function, because the function name itself is the name of the output port.

ACTION: Remove the `output` or `inout` port argument, so the Function Declaration shows only `input` ports, as shown in the following example, which implements a two-input `AND` function:

```
function my_and_func;
        input a;
        input b;
        begin
                my_and_func = a & b;
        end
endfunction
```

To use this function in a Verilog HDL module, you would use the following syntax:

```
assign out = my_func(in1, in2);
```

See also:

Section 10.3 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Gate Instantiation error at <*location*>: gate requires <*number*> inputs and outputs

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used a Gate Instantiation for a predefined logic gate; however, the logic gate is not using the expected number of input and output ports. The gate requires the specified number of connections to function correctly in the design.

ACTION: Check the port syntax in the Gate Instantiation and adjust the connection list accordingly.

See also:

Section 7.1 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Gate Instantiation error at <*location*>: gate requires at least one input and one output port

---

CAUSE: In a Gate Instantiation at the specified location in a Verilog Design File (**.v**), you instantiated a built-in logic gate; however, the gate is missing required ports. Basic logic gates require at least one input and one output connection to function correctly in the circuit.

ACTION: Add the missing input or output ports to the gate.

See also:

Section 7.1 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Gate Instantiation error at <*location*>: multiple terminals on pullup or pulldown sources are not supported

---

CAUSE: In a Gate Instantiation at the specified location in a Verilog Design File (**.v**), you specify a `pullup` or `pulldown` source that drives more than one net in its terminal list. Although Verilog HDL supports `pullup` and `pulldown` sources driving multiple nets, the MAX+PLUS II Advanced Synthesis software allows `pullup` and `pulldown` sources to drive only one net.

ACTION: Rewrite the design to include a separate `pullup` or `pulldown` source for each net that you wish to drive. For example, if you are using the following syntax:

```
pullup my_pullup (a, b, c, d);
```

you should replace it with the following syntax:

```
pullup my_pullup1 (a);
pullup my_pullup2 (b);
pullup my_pullup3 (c);
pullup my_pullup4 (d);
```

See also:

Sections 7.1 and 7.8 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Gate Instantiation error at <*location*>: name-based port connection is not allowed in Gate Instantiation

---

CAUSE: In a Gate Instantiation at the specified location in a Verilog Design File (**.v**), you specified connections by using named ports. The MAX+PLUS II Advanced Synthesis software does not support name-based port connections in Gate Instantiations.

ACTION: Edit the design to make sure the output ports are the first argument in the Gate Instantiation, followed by the input ports, as shown in the following example:

```
g = xor(out, in1, in2)
```

See also:

Section 7 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Gate Instantiation error at <*location*>: unconnected terminal in instantiation of gate <*name*>

CAUSE: In a Gate Instantiation at the specified location in a Verilog Design File (**.v**), you instantiated the specified gate but left one of the terminals unconnected. This error could also be caused by accidentally using two commas instead of one.

ACTION: Connect all the terminals in the Gate Instantiation, and remove any accidental duplicate commas.

See also:

Section 7 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Generate Statement error at <*location*>: variable <*name*> is not declared as genvar

---

CAUSE: In a Generate Statement at the specified location in a Verilog Design File (**.v**), you used a name that is not is not declared as a Generate Statement index variable, or `genvar`. An index variable that is used to control a Generate Statement must be declared as a `genvar`. You cannot use other data types as Generate Loop variables.

ACTION: Make sure each variable used to control a Generate Statement is declared as a `genvar`.

See also:

Section 12.1.3 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Macro Definition error at <*location*>: illegal macro parameters near <*name*>

CAUSE: In a Macro Definition Verilog Design File (**.v**) at the specified location, you specified parameters for the macro, but the parameters are not legal parameters.

ACTION: Make sure you use legal characters and correct syntax for variables, numbers, and other text in the macro parameters.

See also:

Section 19.3 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Macro Definition syntax error at <*location*>: illegal character in macro parameter near <*name*>

CAUSE: In a Macro Definition at the specified location in a Verilog Design File (**.v**), you specified parameters for a macro, but the macro parameter contains illegal characters.

ACTION: Make sure you use legal characters and correct syntax for variables, numbers, and other text in the macro parameters.

See also:

Section 19.3 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Module Declaration error at <*location*>: can't override parameters -- module does not expect any parameters

---

CAUSE:	In a Module Declaration at the specified location in a Verilog Design File (**.v**), you specified a Module Instance Parameter Value Assignment to override one or more specified paramters for the instance; however, no parameters have been defined for the module.
ACTION:	Edit the Module Declaration to define parameters, or edit or remove the parameter overrides that are specified in the Module Instance Parameter Value Assignment.

See also:

Section 12.1 and 12.2 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Module Declaration error at <*location*>: port <*name*> is declared more than once

---

CAUSE: In a Module Declaration at the specified location in a Verilog Design File (**.v**), you declared a port of a module more than once. Ports must be declared only once in a Module Declaration.

ACTION: Delete the extra Port Declaration.

See also:

Sections 2.8, 12.1, and 12.3.3 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Module Declaration error at <*location*>: port <*name*> is not declared as port

---

CAUSE: In a Module Declaration at the specified location in a Verilog Design File (**.v**), you specified a port name, but the specified port name is not declared as a port.

ACTION: Declare the port as an input, output, or bidirectional port.

See also:

Section 12.1 and 12.3 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Module Declaration error at <*location*>: top module port <*name*> is not found in the port list

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you referenced the specified port name for the top-level module, but the specified port name is not found in the list of ports in the top-level module's Module Declaration.

ACTION: Edit the design to make sure the port name appears in the list of ports in top-level module's Module Declaration.

See also:

Section 12.1 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Module Definition error at <*location*>: port <*name*> is defined more than once

---

CAUSE: In a Module Declaration at the specified location in a Verilog Design File (**.v**), you defined two ports with the same name. You must define ports with unique names.

ACTION: Delete the extra Port Definition or give the extra port a unique name.

See also:

Sections 12.3.1 and 12.3.2 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Module Definition error at <*location*>: too many parameters for module instance

---

CAUSE: In a Module Instantiation at the specified location in a Verilog Design File (**.v**), you specified parameters for a module, but the number of parameters specified in the Module Instantiation is more than the number of parameters specified in the Module Definition.

ACTION: Make sure the number of parameters in the Module Instantiation does not exceed the number of parameters in the Module Definition.

See also:

Section 12.1 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Module Instantiation error at <*location*>: too many ports used in Module Instantiation

---

CAUSE: In a Module Instantiation at the specified location in a Verilog Design File (**.v**), you used more ports than the number of ports specified in the Module Definition.

ACTION: Make sure the Module Instantiation uses no more ports than the number of ports specified in the Module Defintion.

See also:

Section 12.1 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Module Instantiation error at <*location*>: arrays of Module Instantiations are not supported

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you instantiated an array of Module Instantiations. Although arrays of Module Instantiations are supported in Verilog HDL, they are not supported in the MAX+PLUS II Advanced Synthesis software.

ACTION: Rewrite the Verilog Design File to implement an array of modules using a Generate Statement instead of an array of Module Instantiations. For example, the following sample design instantiates eight two-input `AND` gates using an array of Module Instantiations:

```
module my_and(in_a, in_b, out);
    input in_a, in_b;
    output out;
    assign out = in_a & in_b;
endmodule
module my_and_array (in1, in2, out);
    input [7:0] in1, in2;
    output [7:0] out;
    my_and and_array[7:0] (in1, in2, out);
endmodule
```

If this design is rewritten with a Generate Statement, as shown in following sample design, the design will then compile successsfully.

```
module my_and(in_a, in_b, out);
    input in_a, in_b;
    output out;
    assign out = in_a & in_b;
endmodule
module my_and_array (in1, in2,out);
    input [7:0] in1, in2;
    output [7:0] out;
    genvar i;
    generate for (i = 0; i<8; i=i+1) begin:my_label_for_each_bit
          my_and and_array(in1[i], in2[i], out[i]);
    end
    endgenerate
endmodule
```

See also:

Sections 7.1 and 12.1 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Module Instantiation error at <*location*>: ignoring trailing ordered association

---

CAUSE: In a Module Instantiation at the specified location in a Verilog Design File (**.v**), you either accidentally entered duplicate commas at the end of the port connection list, or you used named port connections followed by ordered port connections; port connections must be all by order or all by name; the two types cannot be mixed. Integrated Synthesis is ignoring the duplicate commas or ordered port connections.

ACTION: No action is required. To avoid receiving this message in the future, remove any duplicate commas, and make sure the Module Instantiation does not include mixed ordered and named port connections.

See also:

Section 12.3 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Module Instantiation error at <*location*>: module instance port connections cannot be mixed -- port connections must be all by order or all by name

---

CAUSE:	In a Module Instantiation at the specified location in a Verilog Design File (**.v**), you instantiated a module, but specified some of the port connections in ordered form, and others in named form. Port connections must be all by order or all by name; the two types cannot be mixed.
ACTION:	Connect the ports in the Module Instantiation either all by name or all by order.

See also:

Section 12.3.5 and 12.3.6 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Module Instantiation warning at <*location*>: excess data bits passed to module ports will be ignored

---

CAUSE: In a Module Instantiation at the specified location in a Verilog Design File (**.v**), you specified data bits to be passed to module ports, but you specified more data bits than the actual port width read by the MAX+PLUS II Advanced Synthesis software.

ACTION: No action is required. To avoid receiving this message in the future, make sure the number of data bits specified to be passed to module ports does not exceed the actual port width, or make sure the first Module Instantiation for the module instantiates the widest port width.

See also:

Section 12.1 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Module Instantiation warning at <*location*>: instantiated undefined module <*name*>

CAUSE: In a Module Instantiation at the specified location in a Verilog Design File (**.v**) you attempted to instantiate the specified module, but the module is undefined. This message may occur if you do not include the design file that defines the module.

ACTION: No action is required. To avoid receiving this message in the future, make sure you include the design file that defines the module.

See also:

Section 12.1 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Net Declaration error at <*location*>: variable <*name*> was previously declared as reg data type

---

CAUSE: In Net Declaration at the specified location in a Verilog Design File (**.v**), you declared the specified variable as a net data type; however, the variable was previously declared as a reg data type in a Reg Declaration. The variable must be declared as only one data type.

ACTION: Remove either the Net Declaration or the Reg Declaration.

See also:

Sections 3.2.1 and 12.3.3 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Net Declaration error at <*location*>: variable <*name*> was previously declared as single-bit port

---

CAUSE: In a Net Declaration at the specified location in a Verilog Design File (**.v**), you used the specified variable, but the variable was originally declared as a single-bit port and has been redeclared to be a multibit net or reg data type port.

ACTION: No action is required. To avoid receiving this message in the future, change either the Port Declaration or the Net Declaration so both declarations have an equal number of bits for the variable.

See also:

Sections 3.2.1 and 12.3.3 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Net Declaration or Register Declaration error at <*location*>: array of vectors of multibit values is not supported

---

CAUSE: In a Net Declaration or Register Declaration at the specified location in a Verilog Design File (**.v**), you specified an array of vectors of multibit values, such as integers; for example, `integer [0:3] int_array_vec [0:15]`. Verilog HDL does not support arrays of vectors of multibit values.

ACTION: Change the array of vectors of multibit values into a multidimensional array. For example, if you have `integer [0:3] int_array_vec [0:15]`, you can replace it with `integer int_array_vec [0:3][0:15]`.

See also:

Section 3.3 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Net Declaration or Register Declaration error at <*location*>: vector of multibit values is not supported

---

CAUSE: In a Net Declaration or Register Declaration at the specified location in a Verilog Design File (**.v**), you specified a vector of multibit values, such as a vector of integers like `integer [0:3] int_vec`. Verilog HDL does not support vectors of multibit values.

ACTION: Change the multibit vector into an array. For example, you could change `integer [0:3] int_vec` into `integer int_vec [0:3]`.

See also:

Section 3.3 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL or VHDL error at <*location*>: <*text*>

CAUSE: Integrated Synthesis generated the specified error for the specified location in a Verilog Design File (**.v**) or VHDL Design File (**.vhd**).

ACTION: Edit the file to correct the error. A future version of the MAX+PLUS II Advanced Synthesis software will provide more detailed information about this error.

Verilog HDL or VHDL error: *<text>*

CAUSE: Integrated Synthesis generated the specified error for a Verilog Design File (**.v**) or VHDL Design File (**.vhd**).

ACTION: Edit the file to correct the error. A future version of the MAX+PLUS II Advanced Synthesis software will provide more detailed information about this error.

Verilog HDL or VHDL information at <*location*>: <*text*>

CAUSE: Integrated Synthesis generated the specified information for the specified location in a Verilog Design File (**.v**) or VHDL Design File (**.vhd**).

ACTION: No action is required. A future version of the MAX+PLUS II Advanced Synthesis software will provide more detailed information about this information message.

Verilog HDL or VHDL information: *<message description>*

CAUSE: Integrated Synthesis generated the specified information for a Verilog Design File (**.v**) or VHDL Design File (**.vhd**).

ACTION: No action is required. A future version of the MAX+PLUS II Advanced Synthesis software will provide more detailed information about this information message.

Verilog HDL or VHDL unsupported feature warning: multidimensional array name <*name*> not fully supported

---

CAUSE: In a Verilog Design File (**.v**) or VHDL Design File (**.vhd**) you used the specified multidimensional array name. However, the MAX+PLUS II Advanced Synthesis software does not fully support multidimensional arrays. Although synthesis will not be affected, some errors may result when output files are generated.

ACTION: No action is required. To avoid receiving this message in the future, edit the design to use a one- or two-dimensional array.

Verilog HDL or VHDL warning at <*location*>: <*text*>

CAUSE: Integrated Synthesis generated the specified warning for the specified location in a Verilog Design File (**.v**) or VHDL Design File (**.vhd**).

ACTION: Edit the file to avoid problems in the future processing of the design, or to avoid receiving this message in the future. A future version of the MAX+PLUS II Advanced Synthesis software will provide more detailed information about this warning.

Verilog HDL or VHDL warning: *<text>*

CAUSE: Integrated Synthesis generated the specified warning for a Verilog Design File (**.v**) or VHDL Design File (**.vhd**).

ACTION: Edit the file to avoid problems in the future processing of the design, or to avoid receiving this message in the future. A future version of the MAX+PLUS II Advanced Synthesis software will provide more detailed information about this warning.

Verilog HDL Port Declaration warning at <*location*>: port <*name*> was previously declared as array port

---

CAUSE: In a Port Declaration at the specified location in a Verilog Design File (**.v**), you declared the specified port as a module port, but the port was previously declared as an array port. The MAX+PLUS II Advanced Synthesis software is ignoring the second declaration of the port.

ACTION: No action is required. To avoid receiving this message in the future, make sure the Port Declarations are consistent.

See also:

Sections 12.1 and 12.3 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Procedural Assignment error at <*location*>: illegal Procedural Assignment to nonregister data type <*name*>

CAUSE: In a Procedural Assignment at the specified location in a Verilog Design File (**.v**), the left-hand side (LHS) is not a reg, integer, or time data type. The LHS of a Procedural Assignment must be a reg, integer, or time data type.

ACTION: Change the target of the assignment to be a `reg` data type, if that is appropriate, or move the Procedural Assignment out of the enclosing procedural block to make it a Continuous Assignment.

See also:

Sections 6.2 and 9.2 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Reg Declaration error at <*location*>: variable <*name*> was previously declared as net data type

---

CAUSE: In a Reg Declaration at the specified location in a Verilog Design File (**.v**), you declared the specified variable as a reg data type; however, the variable was previously declared as a net data type in a Net Declaration. The variable must be declared as only one data type.

ACTION: Remove either the Net Declaration or the Reg Declaration.

See also:

Sections 3.2.1 and 12.3.3 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL syntax error at <*location*>: illegal character <*character*> in user-defined primtive table

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used a User-Defined Primitive (UDP) Declaration; however, the UDP table contains one or more illegal characters.

ACTION: Make sure the UDP table contains no illegal characters.

See also:

Section 8 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL syntax error at <*location*>: illegal character in binary number

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, a binary number contains an illegal character.

ACTION: Make sure you use only `0` and `1` digits in binary numbers.

See also:

Section 2.5 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL syntax error at <*location*>: illegal character in decimal number

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, a decimal number contains an illegal character.

ACTION: Make sure you use only digits `0` through `9` in decimal numbers.

See also:

Section 2.5 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL syntax error at <*location*>: illegal character in hexadecimal constant value

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, a hexadecimal constant value contains illegal characters, that is, characters other than `0..9`, `a..f`, `x`, or `z`.

ACTION: Make sure the constant value contains only `0..9`, `a..f`, `x`, or `z` characters.

See also:

Section 2.5 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL syntax error at <*location*>: illegal character in octal constant value

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, the specified octal constant value contains one or more illegal characters, that is, characters other than `0..7`, `x`, or `z`.

ACTION: Make sure the octal constant value contains only `0..7`, `x`, or `z` characters.

See also:

Section 2.5 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL syntax error at <*location*>: parameter value must be constant expression

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you specified a value for a module parameter that is not a constant expression; however, parameter values must be constant expressions.

ACTION: Edit the design so the values you assign to parameters are constant expressions.

See also:

Sections 3.11 and 12.2 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL syntax error at <*location*>: syntax error near text '<*text*>'

CAUSE: In a Verilog Design File (**.v**) at the specified location, a syntax error occurred near the specified text. For example, this error may occur if required punctuation, such as a semicolon or parenthesis, is missing before the specified text.

ACTION: Check for and fix any syntax errors that appear immediately before or at the specified keyword.

See also:

Section 6.2.1 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL syntax error at <*location*>: table entry is missing colon ':' I/O separator

---

CAUSE: In a User-Defined Primitive (UDP) Definition at the specified location in a Verilog Design File (**.v**), one of the lines of the UDP table definition is missing the colon character (`:`) that separates inputs from outputs.

ACTION: Insert a colon (`:`) between the inputs and outputs in the table.

See also:

Section 8 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL syntax error at <*location*>: unexpected end of file in If Statement

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used an If Statement that is not ended correctly. For instance, you may have forgotten the end that should appear at the end of the If Statement.

ACTION: Make sure the If Statement is syntactically correct.

See also:

Section 9.4 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL syntax error at <*location*>: vector size reaches 2**<*number*> bits limit

---

CAUSE:	In a Verilog Design File (**.v**) at the specified location, you used the specified vector size, which is greater than or equal to the limit of 2**20 bits.
ACTION:	Reduce the vector size to less than 2**20 bits.

See also:

Sections 3.3 and 3.10 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL syntax error: syntax error near end of file

CAUSE: In a Verilog Design File (**.v**), a syntax error occurred near the end of the file. For example, this error may occur if required punctuation, such as a semicolon or parenthesis, is missing at the end of the file.

ACTION: Check for and fix syntax errors at the end of the file.

Verilog HDL syntax warning at <*location*>: extra block comment delimiter characters (slash and asterisk) within block comment

---

CAUSE:	In a Verilog Design File (**.v**) at the specified location, a line contains extra block comment delimiter characters (that is, a slash and an asterisk, or `/*`), within a `/* */`-style or block comment. Integrated Synthesis is ignoring the extra `/*` characters.
ACTION:	No action is required. To avoid receiving this message in the future, check the comment text for extra `/*` characters and remove them.

See also:

Section 2.3 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Task Definition error at <*location*>: task <*name*> is not used as a task

---

CAUSE: In a Task Definition at the specified location in a Verilog Design File (**.v**), you defined a task that is not being enabled or run correctly. For example, the task may have been called as a function; however, functions have a return value, while tasks do not.

ACTION: Make sure the Task Enable Statement has the correct syntax, with no return value, or change the Task Definition.

See also:

Section 12.1 and 12.3 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Task Enable Statement error at <*location*>: incorrect number of arguments for Task Enable Statement

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used a Task Enable Statement, but the Task Enable Statement does not pass the correct number of arguments for the task. The number of arguments passed must match the number of arguments in the Task Definition exactly.

ACTION: Edit the design to make sure the argument list in the Task Enable Statement matches the argument list in the Task Definition.

See also:

Section 10.2 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL unsupported feature error at <*location*>: bidirectional pass switch gate primitive <*name*> is not supported

---

CAUSE: In a Gate Instantiation at the specified location in a Verilog Design File (**.v**), you instantiated a basic gate primitive that is a `tranif0`, `tranif1`, `rtranif0` or `rtranif1` bidirectional pass switch primitive. However, bidirectional pass switch gate primitives are not supported by the MAX+PLUS II Advanced Synthesis software.

ACTION: Edit the design to replace an enabled bidirectional pass gate with two unidirectional enabled pass gate. For example, you can replace one `tranif1` gate with two `cmos` pass gates.

See also:

Section 7 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL unsupported feature error at <*location*>: Deassign Statement is not supported

---

CAUSE: In an Always Construct at the specified location in a Verilog Design File (**.v**), you used a Deassign Statement; however, although Verilog HDL supports Deassign Statements, which release registers from Procedural Continuous Assignments, they are not supported in the MAX+PLUS II Advanced Synthesis software.

ACTION: Remove both the Procedural Continuous Assignment Statement and the Deassign Statement from the Always Construct and replace them with an If Statement that has appropriate condition expressions for specifying assignment conditions.

See also:

Section 6.2 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL unsupported feature error at <*location*>: Event Trigger Statement is not supported

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used an Event Trigger Statement. Although Verilog HDL supports Event Trigger Statements, they are not supported by Integrated Synthesis.

ACTION: Remove the Event Trigger Statement from the design. If the Event Trigger Statement is being used to trigger an Always Construct in another module, you can combine the two modules to eliminate the need for an Event Trigger Statement.

See also:

Section 9.7 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL unsupported feature error at <*location*>: Event Trigger Statement is not supported

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used an Event Trigger Statement. Although Verilog HDL supports Event Trigger Statements, they are not supported by Integrated Synthesis.

ACTION: Remove the Event Trigger Statement from the design. If the Event Trigger Statement is being used to trigger an Always Construct in another module, you can combine the two modules to eliminate the need for an Event Trigger Statement.

See also:

Section 9.7 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL unsupported feature error at <*location*>: multidimensional array indexing is not supported

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you specified a multidimensional array index. Although multidimensional arrays are supported in Verilog HDL, the MAX+PLUS II Advanced Synthesis software does not support multidimensional arrays.

ACTION: Edit the design to remove the use of multidimensional arrays. The MAX+PLUS II Advanced Synthesis software does support arrays of registers, so you may be able to rewrite your multidimensional array as a register array; for example, you could rewrite the declaration `reg my_array[7:0][7:0]` as `reg [7:0]my_array[7:0]`.

See also:

Sections 3.10 and 26.6 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL unsupported feature error at <*location*>: Procedural Continuous Assignment to register is not supported

---

CAUSE: In an Always Construct, at the specified location in a Verilog Design File (**.v**), you used a Procedural Continuous Assignment Statement to specify an assignment to a register. However, although the Procedural Continuous Assignment Statement, which overrides a regular assignment, is supported in Verilog HDL, it is not supported in the MAX+PLUS II Advanced Synthesis software.

ACTION: Either remove the Procedural Continuous Assignment Statement, or try to merge the assignment into other Always Constructs that contain assignments to the same register, by using an If Statement to specify the assignment conditions instead of using a Procedural Continous Assignment Statement.

See also:

Section 5.6 and 6.2 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL unsupported feature error at <*location*>: real numbers are not supported

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used a real number. Although real numbers are supported in Verilog HDL, the MAX+PLUS II Advanced Synthesis software does not support synthesis of real numbers.

ACTION: Edit the design to use integers instead of real numbers.

See also:

Section 2.5 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL unsupported feature error at <*location*>: real variable data type values are not supported

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you declared a real variable data type. Although Verilog HDL supports real variable data types, this type is not supported in the MAX+PLUS II Advanced Synthesis software.

ACTION: Change the data type of the variable to something other than `real`.

See also:

Section 3.9 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL unsupported feature error at <*location*>: Release Statement not supported for processing with Integrated Synthesis

---

CAUSE:	In a Verilog Design File (**.v**) at the specified location, you used a Release Statement. Although Release Statements are supported in Verilog HDL, they are not supported for processing with Integrated Synthesis. A Release Statement is used in conjuction with a Force Statement to override values on wires or registers. It is typically used with a simulator during design debugging to change signal values; however, it is not supported for synthesis, because you should never need to release the value on a wire or in a net in a synthesized design.
ACTION:	Edit the design to remove the Release Statement.

See also:

Sections 5.6.2 and 9.3.2 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL unsupported feature error at <*location*>: system Function Call <*name*> is not supported

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used a system Function Call to the specified function; however, Integrated Synthesis does not support the system Function Call.

ACTION: Remove the specified Function Call.

See also:

Section 10.3 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL unsupported feature error at file <*name*> (line <*number*>): cannot synthesize MOS switch gate primitive

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you instantiated a `nmos`, `pmos`, or `cmos` MOS switch gate primitive. However, the MAX+PLUS II Advanced Synthesis software does not support synthesis of MOS switch gate primitives.

ACTION: Edit the design to remove all MOS switch gate primitives. You can replace them with behavioral models of the basic gates, or you can rewrite the design in a behavioral style.

See also:

Sections 7.1, 7.5, and 7.7 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL unsupported feature error at file <*name*> line (<*number*>): can't synthesize pullup or pulldown primitive

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used a `pullup` or `pulldown` primitive. However, the `pullup` and `pulldown` gates are not supported for Integrated Synthesis, because resisitve `pullup` and `pulldown` primitives cannot be implemented in Altera devices, except as part of a tri-state driver driving a bidirectional pin.

ACTION: If you need to implement a tri-state driver driving a bidirectional pin, use a construct like the one shown in the following example:

```
if (enable == 1'b1)
begin   tri_out <= my_sig;
else
   tri_out <= 1'bz;
end
```

For all other cases, edit the design to use only logic value `1` and logic value `0`.

See also:

Section 7.1 and 7.8 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL unsupported feature warning at <*location*>: bidirectional pass switch gate primitive <*name*> is not supported and will be implemented as wire

---

CAUSE:	In a Verilog Design File (**.v**) at the specified location, you used the specified bidirectional pass switch gate primitive (`tran` or `rtran`). However, bidirectional pass switch gate primitives are not supported by the MAX+PLUS II Advanced Synthesis software, so Integrated Synthesis will implement the primitive as a wire.
ACTION:	No action required. To avoid receiving this message in the future, edit the design to replace an enabled bidirectional pass gate with two unidirectional enabled pass gates.

See also:

Section 7 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL unsupported feature warning at <*location*>: Disable Statement is not supported, and is ignored in compilation

---

CAUSE:	In a Block Statement at the specified location in a Verilog Design File (**.v**), you used a Disable Statement. However, although Disable Statements are supported in Verilog HDL, they are not supported in the MAX+PLUS II Advanced Synthesis software. The Compiler is ignoring the Disable Statement.
ACTION:	No action is required. To avoid receiving this message in the future, use an If Statement to specify condition expressions that control the termination condition inside the Block Statement.

See also:

Sections 9.6, 9.7, 9.8, and 11 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL unsupported feature warning at <*location*>: ignoring unsupported net type <*name*>

CAUSE: In a Net Declaration at the specified location in a Verilog Design File (**.v**), you used a net type that is not supported by the MAX+PLUS II Advanced Synthesis software. Integrated Synthesis is ignoring the Net Declaration.

ACTION: No action required. To avoid receiving this message in the future, remove the net type or use a net type that is supported by the MAX+PLUS II Advanced Synthesis software.

See also:

Section 3.7 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL unsupported feature warning at <*location*>: Initial Construct is not supported and will be ignored

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used an Initial Construct to specify initial circuit conditions for simulation. Although Verilog HDL supports Initial Constructs, Integrated Synthesis does not support Initial Constructs, and will ignore the associated assignments.

ACTION: No action is required. To avoid receiving this message in the future, check the circuit to make sure the undefined initial state will not cause a functional failure. If necessary, use a reset signal to put the circuit into the desired initial state.

See also:

Sections 9.1 and 9.9 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL unsupported feature warning at <*location*>: Module Item Declaration is ignored

---

CAUSE: In a Module Item Declaration at the specified location in a Verilog Design File (**.v**), you used a feature or construct that is not supported by the MAX+PLUS II Advanced Synthesis software.

ACTION: No action is required. To avoid using this message in the future, remove the unsupported construct.

See also:

The IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL unsupported feature warning at <*location*>: Parallel (Fork-Join) Block is not supported

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used a Parallel Block, which is a block surrounded by Fork and Join Statements. Although Verilog HDL supports Parallel Blocks, the MAX+PLUS II Advanced Synthesis software does not support Parallel Blocks, and ignores all statements between the Fork and Join Statements.

ACTION: Edit the design to remove the Parallel Blocks. You can replace Parallel Blocks with one or more Sequential Blocks, possibly inside separate Always Constructs.

See also:

Section 9.8.2 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL unsupported feature warning at <*location*>: system timing check ignored

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used a system timing check. The MAX+PLUS II Advanced Synthesis software does not support system timing checks, so any system timing checks you specify will be ignored.

ACTION: No action is required. To avoid receiving this message in the future, remove the system timing check.

See also:

Section 15 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL unsupported feature warning at <*location*>: Wait Statement is not supported and is ignored

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you have used a Wait Statement. Although Verilog HDL supports Wait Statements, they are not supported by the MAX+PLUS II Advanced Synthesis software and are ignored.

ACTION: No action is required. To avoid receiving this message in the future, remove the Wait Statement.

See also:

Section 9.7 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL User-Defined Primitive (UDP) Declaration error at <*location*>: sequential table entry <*text*> found in combinational table

---

CAUSE:	In a User-Defined Primitive Declaration at the specified location in a Verilog Design File (**.v**), you used a UDP table; however, the UDP table, which Integrated Synthesis has identified as modeling combinational logic, has a table entry for modeling sequential logic.
ACTION:	Change the UDP table to be either entirely combinational, or entirely sequential.

See also:

Section 8.1 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL User-Defined Primitive Declaration error at *<location>*: incorrect number of inputs for UDP table entry *<text>*

CAUSE:	In a User-Defined Primitive (UDP) Declaration at the specified location in Verilog Design File (**.v**), you entered an incorrect number of inputs for the specified UDP table entry. The number of inputs in the UDP table entry must match the number of input ports in the primitive.
ACTION:	Edit either the UDP table entry or the UDP input port list so that the number of input fields in the UDP table matches the number of input ports in the primitive.

See also:

Section 8.1 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL User-Defined Primitive Declaration error at *<location>*: incorrect output field length in UDP table *<name>*

CAUSE:	In a User-Defined Primitive (UDP) Declaration at the specified location in Verilog Design File (**.v**), you entered an incorrect output field length in a UDP table. A UDP table can have only a single 1-bit output signal.
ACTION:	Correct the table output so it has only a single 1-bit output signal.

See also:

Section 8.1 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL User-Defined Primitive Declaration error at <*location*>: inout ports are not allowed on UDP

CAUSE: In a User-Defined Primitive (UDP) Declaration at the specified location in a Verilog Design File (**.v**), you specified an `inout` port; however, a UDP cannot use the `inout` port type. Only `input` and `output` ports are allowed for UDPs.

ACTION: Edit the UDP port declarations so they specify only `input` and `output` ports.

See also:

Section 8.1 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL User-Defined Primitive Declaration error at <*location*>: missing present state field in UDP table <*text*>

CAUSE: In a User-Defined Primitive (UDP) Declaration at the specified location in a Verilog Design File (**.v**), you used a UDP sequential table, but the UDP table is missing the present state field, which is required.

ACTION: Edit the design to include a present state field in the UDP table, in the following format:

```
[inputs : present state : next state]
```

See also:

Section 8 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL User-Defined Primitive Declaration error at <*location*>: rising or falling edge is not allowed in combinational user-defined primitive <*name*>

CAUSE:	In the specified User-Defined Primitive (UDP) Declaration at the specified location in a Verilog Design File (**.v**), you used a rising or falling edge; however, rising or falling edges are not allowed in combinational UDPs.
ACTION:	Edit the design to remove the rising or falling edge from the specified UDP.

See also:

Section 8 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL User-Defined Primitive Declaration error at <*location*>: user-defined primitive has no output ports, but must have exactly one output port

---

CAUSE:	In a User-Defined Primitive (UDP) Declaration at the specified location in a Verilog Design File (**.v**), you specified a UDP with no output ports; however, a UDP must have exactly one output port.
ACTION:	Add a single output port to the UDP.

See also:

Section 8 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL User-Defined Primitive Declaration error at <*location*>: user-defined primitive must have only one output port

---

CAUSE:	In a User-Defined Primitive (UDP) Declaration at the specified location in a Verilog Design File (**.v**), you specified a UDP with more than one output port; however, a UDP must have only one output port.
ACTION:	Make sure the UDP has only one output port.

See also:

Section 8 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL User-Defined Primitive Declaration warning at <*location*>: user-defined primitive table is empty

---

CAUSE: In a User-Defined Primitive (UDP) Declaration at the specified location in a Verilog Design File (**.v**), you used a UDP table that is empty. As a result, Integrated Syntheses will create an empty primitive gate.

ACTION: No action is required. To avoid receiving this message in the future, enter the correct logic function information into the UDP table.

See also:

Section 8 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL User-Defined Primitive error at <*location*>: user-defined primitive port must be scalar port

---

CAUSE: In a User-Defined Primitive (UDP) Declaration at the specified location in a Verilog Design File (**.v**), you declared a vector port; however, all UDP ports must be scalar.

ACTION: Replace the vector port with multiple scalar ports.

See also:

Section 8 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Variable Declaration error at <*location*>: variable <*name*> is already declared

---

CAUSE: In a Variable Declaration at the specified location in a Verilog Design File (**.v**), you declared a variable that had already been declared in the design. You must declare a variable only once in a design.

ACTION: Remove the duplicate Variable Declaration from the design.

See also:

Sections 3.2.2 and 6.2.1 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Variable Declaration error at <*location*>: variable <*name*> is already declared as reg data type

---

CAUSE: In a Variable Declaration at the specified location Verilog Design File (**.v**), you declared a variable as a reg data type, but the variable had already been declared as a reg data type. You must declare a variable only once.

ACTION: Remove any extra Variable Declarations for the variable.

See also:

Sections 3.2.2 and 6.2.1 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Variable Declaration error at <*location*>: variable <*name*> is already declared as wire data type

---

CAUSE: In a Variable Declaration at the specified location in a Verilog Design File (**.v**), you declared the specified variable as a wire data type, but the variable had already been declared as a wire data type. You must declare a variable only once in a design.

ACTION: Remove any extra Variable Declarations for the variable.

See also:

Sections 3.2.2 and 6.2.1 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL Variable Declaration error at <*location*>: variable name <*name*> is already used

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you declared the specified variable name, but the name is already defined elsewhere in the design. For example, you may have used the same name for a wire and for a module instantiation. All identifiers within a single module must have unique names.

ACTION: Choose a new name for the variable.

Verilog HDL warning at <*location*>: ignoring system Task Enable Statement

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used a Task Enable Statement to enable a system task. The MAX+PLUS II Advanced Synthesis software does not support using a Task Enable Statement to enable a system task. As a result, Integrated Synthesis is ignoring the Task Enable Statement.

ACTION: No action is required. To avoid receiving this message in the future, you can remove the Task Enable Statement.

See also:

Section 10.2 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL warning at <*location*>: index of variable <*name*> is not within address range -- return value is undefined (X)

CAUSE:	In a Verilog Design File (**.v**) at the specified location, you attempted to read a value for the specified variable at a bit index that not within the declared range for the address. As a result, the return value is undefined (X).
ACTION:	No action is required. To avoid receiving this message in the future, make sure the index of the variable is within the declared range for the address.

See also:

Section 3.2 and 6.2 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL warning at <*location*>: memory size reaches 2**<*number*> bits

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you specified a memory size; however, the memory size you specified is greater than or equal to 2**24 bits. Although the specified memory size is supported, it is very large.

ACTION: No action is required. To avoid receiving this message in the future, reduce the memory size to less than 2**24 bits if it is appropriate for the design.

See also:

Section 3.10.3 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL warning at <*location*>: port <*name*> was previously declared with different range

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you declared the specified module port with a different range than was previously specified in the design. The MAX+PLUS II Advanced Synthesis software is using the larger of the two ranges.

ACTION: No action is required. To avoid receiving this message in the future, make sure the module Port Declaration ranges match.

See also:

Section 12.3 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

Verilog HDL warning at <*location*>: vector size reaches 2**<*number*> bits

---

CAUSE: In a Verilog Design File (**.v**) at the specified location, you used the specified vector size, which greater than or equal to 2**16 bits. Although the specified vector size is legal, it is very large.

ACTION: No action is required. To avoid receiving this message in the future, reduce the vector size to less than 2**16 bits if it is appropriate for the design.

See also:

Sections 3.3 and 3.10 of the IEEE Std. 1364-2001 *IEEE Standard Verilog Hardware Description Language* manual

VHDL aggregate error at <*location*>: aggregate contains one value for two or more elements, but elements must have same type

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used an aggregate for a record type object. In the aggregate, you specified one value for two or more of the object elements. However, the elements do not all have the same type. For example, the first Signal Assignment Statement in the following code uses the `OTHERS` choice to specify the value `1` for the elements `ex2` and `ex3`. The second Signal Assignment Statement specifies the value 3 for the elements `ex1` and `ex2`. However, `ex1` and `ex3` have an integer type, and `ex2` has a string type.

```
TYPE example IS
   RECORD
      ex1 : INTEGER range 1 to 5;
      ex2 : STRING
      ex3 : INTEGER range 1 to 7;
   END example;
SIGNAL exa : example;

exa <= (ex1 => 3, OTHERS => 1);
exa <= (ex1 | ex2 => 3, ex3 => 2);
```

When you specify a value for two or more object elements, the elements must all have the same type.

ACTION: Make sure all the elements for a value in the aggregate have the same type by changing the values in the aggregate or changing the element types in the Type Declaration. In the example above, you can use the following aggregates, or change `ex2` to have an integer type in the Type Declaration:

```
exa <= (ex1 => 3, ex2 => "1", OTHERS => 1);
exa <= (ex1 => 3,  ex2 => "3", ex3 => 2);
```

See also:

Sections 3.2.2 and 7.3.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL aggregate error at <*location*>: aggregate for array type or record type object must cover all elements of object

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used an aggregate for an array type or record type object. However, the aggregate does not cover all the elements of the object. For example, in the Signal Assignment Statement in the following code, the aggregate `(ex1 => 2, ex3 => 5)` does not cover the element `ex2`.

```
TYPE example IS
   RECORD
      ex1 : INTEGER range 1 to 5;
      ex2 : INTEGER range 1 to 6;
      ex3 : INTEGER range 1 to 7;
   END example;
SIGNAL exa : example;

exa <= (ex1 => 2, ex3 => 5);
```

The aggregate for an array type or record type object must cover all the elements of the object.

ACTION: Add the missing elements to the aggregate, or use the `OTHERS` choice to cover all possible values that are not included in the aggregate. In the example above, you can use one of the following aggregates:

```
(ex1 => 2, ex2 | ex3 => 5);
(ex1 => 2, ex3 | OTHERS => 5);
(ex1 => 2, OTHERS => 5);
```

See also:

Sections 3.2.1, 3.2.2 and 7.3.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL aggregate error at <*location*>: array aggregate cannot have overlapping choices

---

CAUSE: In an array aggregate at the specified location in a VHDL Design File (**.vhd**), you specified a choice that is a discrete range that shares one or more of its elements with another choice in the aggregate. You must specify an element only once in an aggregate.

ACTION: Remove one of the duplicate choices, or change either of the conflicting choices so they do not share elements.

See also:

Section 7.3.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL aggregate error at <*location*>: can't determine type of aggregate -- found <*number*> possible types

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you assigned an aggregate to an object. However, Integrated Synthesis cannot determine the type of the aggregate because it found the specified number of different possible types for the aggregate.

ACTION: Make sure the design clearly specifies the type of the aggregate. For example, check the type of the object to which you assigned the aggregate.

See also:

Sections 7.3.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL aggregate error at <*location*>: choice <*name*> can be used only once in aggregate

---

CAUSE: In an aggregate at the specified location in a VHDL Design File (**.vhd**), you used the specified choice. However, you already used the choice in the aggregate. You can use a choice only once in an aggregate.

ACTION: Remove one of the duplicate choices from the aggregate.

See also:

Section 7.3.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL aggregate error at <*location*>: choice <*name*> must belong to index subtype of array aggregate

---

CAUSE: In an array aggregate at the specified location in a VHDL Design File (**.vhd**), you used the specified choice. However, the choice does not belong to the array aggregate's index subtype. The choice in an array aggregate must belong to the aggregate's index subtype.

ACTION: Change the choice so belongs to the array aggregate's index subtype.

See also:

Section 7.3.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL aggregate error at <*location*>: choice <*name*> must belong to index subtype of array aggregate

---

CAUSE: In an array aggregate at the specified location in a VHDL Design File (**.vhd**), you used the specified choice. However, the choice does not belong to the array aggregate's index subtype. The choice in an array aggregate must belong to the array aggregate's index subtype.

ACTION: Change the choice so it belongs to the array aggregate's index subtype.

See also:

Section 7.3.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL aggregate error at <*location*>: choice <*name*> must have <*number*> elements

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the specified choice. However, the choice does not have the correct number of elements. The choice must have the specified number of elements.

ACTION: Change the choice so it has the correct number of elements.

See also:

Section 8.8 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL aggregate error at <*location*>: choice must be constant

---

CAUSE: In an aggregate at the specified location in a VHDL Design File (**.vhd**), you used a non-constant choice. However, the choice must be constant.

ACTION: Change the choice so it is constant.

See also:

Section 7.3.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL aggregate error at <*location*>: choice must be discrete range

---

CAUSE: In an aggregate at the specified location in a VHDL Design File (**.vhd**), you used a non-discrete range for a choice. However, the choice must be a discrete range.

ACTION: Change the choice so it is a discrete range.

See also:

Section 7.3.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL aggregate error at <*location*>: OTHERS choice in record aggregate must specify at least one element

---

CAUSE: In an aggregate for a record type at the specified location in a VHDL Design File (**.vhd**), you used the `OTHERS` choice. However, the record type does not contain any elements for the `OTHERS` choice to specify. The `OTHERS` choice must specify at least one element in the record type.

ACTION: Remove the `OTHERS` choice from the aggregate, or add another element to the record type.

See also:

Sections 3.2.2 and 7.3.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL aggregate error at <*location*>: OTHERS choice must be last choice in aggregate, and must contain only the OTHERS keyword

---

CAUSE:	In an aggregate at the specified location in a VHDL Design File (**.vhd**), you used the `OTHERS` choice. However, the `OTHERS` choice is not the last choice in the aggregate, or contains elements. The `OTHERS` choice must be the last choice in the aggregate, and must contain only the `OTHERS` keyword.
ACTION:	Change the `OTHERS` choice so it is last choice in the aggregate, and it contains only the `OTHERS` keyword.

See also:

Section 7.3.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL aggregate error at <*location*>: OTHERS choice used in aggregate for unconstrained record or array type is not supported

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the `OTHERS` choice in an aggregate for an unconstrained record or array type. However, Integrated Synthesis does not support the `OTHERS` choice in an unconstrained record or array type aggregate.

ACTION: Make sure you do not use the `OTHERS` choice in an aggregate for an unconstrained record or array type, or make sure the record or array type is constrained when the type's aggregate uses the `OTHERS` choice.

See also:

Section 7.3.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL aggregate error at <*location*>: target aggregate for Variable Assignment Statement cannot contain OTHERS choice

---

CAUSE:	In a VHDL Design File (**.vhd**) at the specified location, you used an aggregate in a Variable Assignment Statement. However, you used the `OTHERS` choice in the aggregate. An aggregate for a Variable Assignment Statement cannot contain the `OTHERS` choice.
ACTION:	Remove the `OTHERS` choice from the aggregate.

See also:

Sections 7.3.2 and 8.5 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Alias Declaration error at <*location*>: alias subtype must have same number of elements as name of the object on which the alias is based

---

CAUSE: In an Alias Declaration at the specified location in a VHDL Design File (**.vhd**), you specified a subtype for an alias. However, the subtype does not have the same number of elements as the name of the object on which the alias is based. The alias subtype and the target object of the alias must have the same number of elements.

ACTION: Change the alias subtype or the object so they have the same number of elements.

See also:

Section 4.3.3 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Alias Declaration error at <*location*>: subtype of alias <*name*> for object must have same bounds and direction as object subtype

---

CAUSE: In an Alias Declaration at the specified location in a VHDL Design File (**.vhd**), you specified an alias for an object. However, the subtype you specified for the alias does not have the same bounds or direction as the subtype for the object. The subtype for the alias and the subtype for the object must have the same bounds and direction.

ACTION: Remove the subtype from the Alias Declaration, or change either the alias subtype or the object subtype so they have the same bounds and direction.

See also:

Section 4.3.3.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Architecture Body warning at <*location*>: secondary unit already exists -- overwriting existing secondary unit with new secondary unit

---

CAUSE:	In an Architecture Body at the specified location in a VHDL Design File (**.vhd**), you declared a secondary unit for a design entity. However, you already declared the secondary unit in the same file, or in a different file for the same design entity. As a result, Integrated Synthesis replaced the existing secondary unit with the new secondary unit.
ACTION:	If you do not want Integrated Synthesis to overwrite the existing secondary unit, or to avoid receiving this message in the future, change the new secondary unit so it has a unique name.

See also:

Section 1.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Assertion Statement information at <*location*>: assertion <*text*> is always false

---

CAUSE: In an Assertion Statement at the specified location in a VHDL Design File (**.vhd**), you defined the specified assertion. However, the assertion is never true.

ACTION: To avoid problems in the future processing of the design, or to avoid receiving this message in the future, change the Assertion Statement or its surrounding logic so the assertion is not always false. If you expect the Assertion Statement to always be false, no action is required.

See also:

Section 8.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Association List error at <*location*>: actual associated with formal <*name*> of mode OUT or BUFFER cannot be in the form of a type conversion

---

CAUSE: In a named or positional Association List at the specified location in a VHDL Design File (**.vhd**), you used an actual that is in the form of a type conversion, where the expression to be converted is the actual designator. However, the formal to which the actual is associated has a mode of `OUT` or `BUFFER`. The actual can be in the form of a type conversion only when the mode of the associated formal is `IN`, `INOUT`, or `LINKAGE`, and the actual is not `OPEN`

ACTION: Change the actual so it is not in the form of a type conversion.

See also:

Section 4.3.2.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Association List error at <*location*>: actual parameter assigned to formal parameter <*name*>, but formal parameter is not declared

---

CAUSE: In an Association List at the specified location in a VHDL Design File (**.vhd**), you used named association to map an actual parameter to the specified formal parameter of an item (such as a component or function). However, you did not declare the formal parameter for the item. For example, the Association List in the following code assigns the actual parameter `i` to the formal parameter `i` of the component `dff`, but the Component Declaration for `dff` does not declare a formal parameter `i`:

```
ENTITY err IS
   PORT
   (
      clk   : IN   BIT;
      i     : IN   BIT;
      o     : OUT  BIT
    );
END err;

ARCHITECTURE a OF err IS
   COMPONENT dff
      PORT (d, clk : IN  BIT;
            q      : OUT BIT);
   END COMPONENT;
BEGIN
   dff1 : dff PORT MAP (i=>i, q=>o, clk=>clk);
END a;
```

You must declare a formal parameter before you assign an actual parameter to it.

ACTION: Assign actual parameters only to formal parameters that are declared for the item. For example, in the Association List above, `i=>i` must be `d=>i`.

See also:

Section 4.2.1.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Association List error at <*location*>: formal <*name*> of mode IN cannot be a Type Conversion or Function Call in named Association List

---

CAUSE: In an Association List at the specified location in a VHDL Design File (**.vhd**), you used a Type Conversion or Function Call as a formal. However, the mode of the formal is `IN` and/or the corresponding actual is the `OPEN` keyword. To use a Type Conversion or Function Call as a formal, the mode of the formal must be `OUT`, `INOUT`, `BUFFER`, or `LINKAGE` and the actual cannot be the `OPEN` keyword.

ACTION: Make sure you use a Type Conversion or Function Call as a formal only when the mode of the formal is `OUT`, `INOUT`, `BUFFER`, or `LINKAGE` and the corresponding actual is not the `OPEN` keyword.

See also:

Section 4.3.2.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Association List error at <*location*>: formal <*name*> that is associated individually cannot be associated with actual of OPEN

CAUSE: In an Association List at the specified location in a VHDL Design File (**.vhd**), you associated the specified formal, which is associated individually, with an actual of `OPEN`. However, you cannot associate a formal that is associated individually with an actual of `OPEN`.

ACTION: Associate the formal with an actual that is not `OPEN`.

See also:

Section 4.3.2.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Association List error at <*location*>: positional associations must be listed before named associations

---

CAUSE: In an Association List at the specified location in a VHDL Design File (**.vhd**), you listed named associations before positional associations. However, you must list all positional associations in an Association List before all named associations.

ACTION: List all positional associations before all named associations.

See also:

Section 4.3.3.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Attribute Declaration error at <*location*>: attribute <*name*> has invalid type

---

CAUSE: In an Attribute Declaration at the specified location in a VHDL Design File (**.vhd**), you specified a type for the specified attribute. However, an attribute cannot have the type you specified.

ACTION: Specify a valid type for the attribute.

See also:

Section 4.4 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL attribute error at <*location*>: attribute <*name*> that is used for multiple bits is not synthesizable

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used an attribute for multiple bits. However, this attribute is not synthesizable for multiple bits.

ACTION: Make sure you use the attribute for single bits only.

See also:

Sections 5.1 and 14 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL attribute error at <*location*>: attribute index must be constant integer

---

CAUSE 1:	In an Attribute Specification at the specified location in a VHDL Design File (**.vhd**), you specified an attribute for an object. However, you used a non-constant integer value as the index for the attribute. The attribute index must be a constant integer value.
ACTION:	Change the attribute index to a constant integer value, or remove the attribute from the object.
CAUSE 2:	In a VHDL Design File (**.vhd**) at the specified location, you specified a predefined attribute for an object. However, you used a non-constant integer value as the index for the attribute. The attribute index must be a constant integer value.
ACTION:	Change the attribute index to a constant integer value, or remove the predefined attribute from the object.

See also:

Sections 5.1 and 14 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL attribute error at <*location*>: name <*name*> was used but not declared as an object

---

CAUSE: At the specified location in a VHDL Design File (**.vhd**), you used the specified name for an item (for example, an attribute). However, the name was not declared as an object, that is, as a signal, variable, or constant. You must declare the name as an object before you can use it for the item.

ACTION: Replace the name with a name that is declared as an object. Or, declare the name as an object.

See also:

Sections 4.4 and 5.1, and 6.6 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL attribute error at <*location*>: object with attribute <*name*> must have correct number of dimensions

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you attempted to use the specified attribute for an object. However, the object does not have the number of dimensions that the attribute requires. For example, an attribute that returns the length of an array requires an array type object, but the object may have a scalar type.

ACTION: Change the attribute or the object so that the number of dimensions the attribute requires matches the number of dimensions the object has.

See also:

Sections 4.4, 5.1, 6.6, and 14 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL attribute error at <*location*>: object with attribute <*name*> must have scalar or array type

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you attempted to use the specified attribute for an object. However, the object does not have an array or scalar type. The attribute requires that the object have an array or scalar type. For example, an attribute that returns the range of an object requires an array or scalar type object, but the object may have a record type.

ACTION: Change the attribute or the object so that the object type the attribute requires matches the type of the object.

See also:

Sections 4.4, 5.1, 6.6, and 14 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL attribute error at <*location*>: predefined attribute <*name*> cannot be used in an expression

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the specified predefined attribute in an expression. However, the predefined attribute cannot be used in an expression.

ACTION: Make sure you do not use the predefined attribute in an expression.

See also:

Sections 7 and 14 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL attribute error at <*location*>: predefined attribute <*name*> cannot be used in an expression

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the specified predefined attribute in an expression. However, the predefined attribute cannot be used in an expression.

ACTION: Make sure you do not use the predefined attribute in an expression.

See also:

Sections 7 and 14 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL attribute error at <*location*>: predefined attribute <*name*> cannot be used in Signal Assignment Statement or Variable Assignment Statement

---

CAUSE:	In a VHDL Design File (**.vhd**) at the specified location, you used the specified predefined attribute in a Signal Assignment Statement or a Variable Assignment Statement. However, the predefined attribute cannot be used in an assignment.
ACTION:	Removed the predefined attribute from the assignment.

See also:

Sections 8.4, 8.5, and 14.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL attribute error at <*location*>: predefined attribute <*name*> must have an argument

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the specified predefined attribute. However, you did not use an argument for the predefined attribute. The predefined attribute must have an argument.

ACTION: Add an argument to the predefined attribute.

See also:

Section 14 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL attribute error at <*location*>: predefined attribute <*name*> must have an integer argument

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the specified predefined attribute. However, the argument you used for the predefined attribute does not have an integer type. The predefined attribute must have an integer argument.

ACTION: Change the argument so it has an integer type.

See also:

Section 14 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL attribute error at <*location*>: predefined attribute <*name*> must specify a range

---

CAUSE: In a VHDL Design File (**.vhd**), you used the specified predefined attribute in the specified location, which requires a range. However, the predefined attribute does not specify a range.

ACTION: Replace the predefined attribute with a predefined attribute that specifies a range. Or, replace the predefined attribute with a range.

See also:

Section 14.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL attribute error at <*location*>: prefix of 'LENGTH attribute must be an object of array type or an array subtype

---

CAUSE:	In a VHDL Design File (**.vhd**) at the specified location, you used a prefix with the `'LENGTH` attribute, but the prefix is not an object of array type or an array subtype.
ACTION:	Make sure the prefix of the `'LENGTH` attribute is an object of array type or an array subtype.

See also:

Section 14.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL attribute error at <*location*>: result of attribute <*name*> is out of range for type <*type*>

CAUSE: At the specified location in a VHDL Design File (**.vhd**), you used the specified attribute for the specified type. However, the result of the attribute exceeds the range of the type. The result must belong to the type range.

ACTION: Change the attribute's parameter so the attribute's result belongs to the range of the type.

See also:

Sections 5.1 and 14.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL attribute error at <*location*>: return value type of attribute <*name*> must match object type <*type*>

CAUSE 1: In an Attribute Specification at the specified location in a VHDL Design File (**.vhd**), you specified an attribute for an object. However, the return value of the attribute has a type that is not the same as the specified type of the object. The return value type and the object type must be the same.

ACTION: Remove the attribute from the object, or change the object type or attribute return value type so they are the same.

CAUSE 2: In a VHDL Design File (**.vhd**) at the specified location, you specified a predefined attribute for an object. However, the return value of the attribute has a type that is not the same as the specified type of the object. The return value type and the object type must be the same.

ACTION: Remove the predefined attribute from the object, or change the object type so it is the same as the attribute return value type.

See also:

Sections 5.1 and 14 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL attribute error at <*location*>: user-defined attribute <*name*> used for, but not associated with, object <*name*>

CAUSE: At the specified location in a VHDL Design File (**.vhd**), you used the specified user-defined attribute for the specified object. However, you did not associate the attribute with the object in an Attribute Specification. You must associate a user-defined attribute with an object before you can use the attribute for the object.

ACTION: Associate the attribute with the object in an Attribute Specification.

See also:

Sections 4.4, 5.1, and 6.6 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Attribute Specification error at <*location*>: attribute <*name*> is used but not declared

---

CAUSE: In an Attribute Specification at the specified location in a VHDL Design File (**.vhd**), you used the specified attribute. However, you did not declare the attribute in an Attribute Declaration. You must declare an attribute before you can use it.

ACTION: Replace the attribute with a previously defined attribute, or define the attribute in an Attribute Declaration. However, do not define an attribute with the name you used in the Attribute Specification if you already used the name for a different construct.

See also:

Sections 4.4 and 5.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Attribute Specification error at <*location*>: Attribute Specification cannot contain both ALL or OTHERS keyword and list of entities

---

CAUSE:	In an Attribute Specification at the specified location in a VHDL Design File (**.vhd**), you specified a list of entities. However, you also specified the `ALL` or `OTHERS` keyword in the Attribute Specification. An Attribute Specification that contains the `ALL` or `OTHERS` keyword cannot contain a list of entities.
ACTION:	Remove the list of entities, or remove the `ALL` or `OTHERS` keyword, from the Attribute Specification.

See also:

Section 5.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Attribute Specification warning at <*location*>: encoding values for ENUM_ENCODING attribute are different lengths -- ignored encoding values

---

CAUSE: In an Attribute Specification at the specified location in a VHDL Design File (**.vhd**), you specified the `ENUM_ENCODING` attribute for an enumeration type object that you declared for a state machine. However, the expression in the Attribute Specification contains encoding values of different lengths. For example, in the following Attribute Specification, the encoding value `"1"` is not the same length as the encoding values `"01"`, `"10"`, and `"00"`:

```
ATTRIBUTE ENUM_ENCODING OF count_state: TYPE IS "1 01 10 00"
```

All the encoding values in an attribute expression must be the same length. As a result, the Compiler ignored all the encoding values in the expression.

ACTION: Make sure all the encoding values in the expression of an Attribute Specification are the same length.

☞ The `ENUM_ENCODING` attribute is MAX+PLUS II-specific, and may not be available with other VHDL tools.

See also:

Manually Specifying State Assignments
Section 5.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Attribute Specification warning in <*location*>: ENUM_ENCODING attribute for object with enumeration type <*name*> contains one or more encoding values that are not valid -- ignored encoding values

---

CAUSE: In an Attribute Specification at the specified location in a VHDL Design File (**.vhd**), you specified the `ENUM_ENCODING` attribute for an object with the specified enumeration type, which you declared for a state machine. However, the expression in the Attribute Specification contains encoding values of different lengths. For example, in the following Attribute Specification, the encoding value `"1"` is not the same length as the encoding values `"01"`, `"10"`, and `"00"`:

```
ATTRIBUTE ENUM_ENCODING OF count_state: TYPE IS "1 01 10 00"
```

All the encoding values in an attribute expression must be the same length. As a result, the Compiler ignored all the encoding values in the expression.

ACTION: Make sure all the encoding values in the expression of an Attribute Specification are the same length.

☞ The `ENUM_ENCODING` attribute is MAX-PLUS II-specific, and may not be available with other VHDL tools.

See also:

Manually Specifying State Assignments
Section 5.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Binding Indication error at <*location*>: design entity <*name*> does not contain generic <*name*> specified in associated component

---

CAUSE: In a Binding Indication at the specified location in a VHDL Design File (**.vhd**), you associated a component with the specified design entity. However, the Component Declaration for the component contains the specified actual generic, which does not exist as a formal generic in the design entity. The generic names and types in the component's Component Declaration must match those in the design entity.

ACTION: Add the generic (with the same name and type) to the design entity, or remove the generic from the component's Component Declaration.

See also:

Sections 4.5 and 5.2.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Binding Indication error at <*location*>: design entity <*name*> does not contain port <*name*> specified in associated component

---

CAUSE: In a Binding Indication at the specified location in a VHDL Design File (**.vhd**), you associated a component with the specified design entity. However, the Component Declaration for the component contains the specified actual port, which does not exist as a formal port in the design entity. The port names and types in the component's Component Declaration must match those in the design entity.

ACTION: Add the port (with the same name and type) to the design entity, or remove the port from the component's Component Declaration.

See also:

Sections 4.5 and 5.2.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Binding Indication error at <*location*>: generic <*name*> in design entity does not have <*type*> type that is specified for the same generic in the associated component

---

CAUSE: In a Binding Indication at the specified location in a VHDL Design File (**.vhd**), you associated a component with a design entity. However, the specified formal generic in the design entity does not have the specified type, which is the type that the component's Component Declaration specifies for the actual generic with the same name. The generic names and types in the component's Component Declaration must match those in the design entity.

ACTION: Change the generic in the design entity or Component Declaration so the generics have the same name and type, or remove the generic from both the design entity and the Component Declaration.

See also:

Sections 4.5 and 5.2.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Binding Indication error at <*location*>: port <*name*> in design entity does not have <*type*> type that is specified for the same generic in the associated component

---

CAUSE: In a Binding Indication at the specified location in a VHDL Design File (**.vhd**), you associated a component with a design entity. However, the specified formal port in the design entity does not have the specified type, which is the type that the component's Component Declaration specifies for the actual port with the same name. The port names and types in the component's Component Declaration must match those in the design entity.

ACTION: Change the port in the design entity or Component Declaration so the ports have the same name and type, or remove the port from both the design entity and the Component Declaration.

See also:

Sections 4.5 and 5.2.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Block Configuration error at <*location*>: block <*name*> is configured but not defined

---

CAUSE: In a Block Configuration at the specified location in a VHDL Design File (**.vhd**), you specified the configuration for the specified block. However, you did not define the block in a Block Statement. You must define a block before you can specify a configuration for the block.

ACTION: In the Block Configuration, use a block that you already defined in a Block Statement. Or, use a Block Statement to define the block that is in the Block Configuration. However, do not define a block with the name you used in the Block Configuration if you already used the name for a different construct.

See also:

Sections 1.3.1 and 9.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Block Configuration error at <*location*>: block <*name*> is configured more than once

---

CAUSE: In a Block Configuration at the specified location in a VHDL Design File (**.vhd**), you specified the configuration for the specified block. However, in a previous Block Configuration, you specified a configuration for the same block. You must specify only one configuration per block.

ACTION: Remove the extra Block Configuration for the specified block.

See also:

Section 1.3.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Block Configuration error at <*location*>: Block Configuration cannot have range or index because block <*name*> does not have range

---

CAUSE: In a Block Configuration at the specified location in a VHDL Design File (**.vhd**), you specified the configuration for the specified block using a range or index. However, you did not specify a range for the block in a Generate Statement. If the block in the Block Configuration does not have a range, the Block Configuration cannot have a range or an index.

ACTION: Remove the range or index from the Block Configuration.

See also:

Sections 1.3.1, 9.1, and 9.7 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Block Configuration error at <*location*>: illegal block specification name

---

CAUSE: In a Block Configuration at the specified location in a VHDL Design File (**.vhd**), you used a block specification name. However, the block specification name is illegal, for example, the block specification name may not be visible at the current scope.

ACTION: Make sure the block specification name is legal, for example, that it is visible at the current scope.

See also:

Sections 1.3.1, 10.2, and 10.3 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Block Configuration error at <*location*>: OPEN keyword cannot be used as index

---

CAUSE: In a Block Configuration at the specified location in a VHDL Design File (**.vhd**), you defined the configuration of a block using an index. However, you used the `OPEN` keyword as the index. You cannot use the `OPEN` keyword as an index in a Block Configuration.

ACTION: Replace the `OPEN` keyword with a legal index for the Block Configuration.

See also:

Section 1.3.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Block Configuration or Component Configuration error at <*location*>: component instance or block <*name*> is used but not defined

---

CAUSE 1: In a Block Configuration at the specified location in a VHDL Design File (**.vhd**), you specified the configuration for the specified block. However, you did not define the block in a Block Statement. You must define a block before you can specify a configuration for the block.

ACTION: In the Block Configuration, use a block that you already defined in a Block Statement. Or, use a Block Statement to define the block that is in the Block Configuration. However, do not define a block with the name you used in the Block Configuration if you already used the name for a different construct.

CAUSE 2: In a Component Configuration at the specified location in a VHDL Design File (**.vhd**), you specified the configuration for the specified component instance. However, you did not define the component instance in a Component Instantiation Statement. You must define a component instance before you can specify a configuration for the instance.

ACTION: In the Component Configuration, use a component instance that you already defined in a Component Instantiation Statement. Or, use a Component Instantiation Statement to define the component instance that is in the Component Configuration. However, do not define a component with the name you used in the Component Configuration if you already used the name for a different construct.

See also:

Sections 1.3.1. 1.3.2, 9.1, and 9.6 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Case Statement error at <*location*>: Case Statement cannot be used as Register Inference because Case Statement cannot contain an edge-triggered or event-triggered condition for the statement's execution. Use If Statement.

CAUSE:	In a VHDL Design File (**.vhd**) at the specified location, you used a Case Statement as a Register Inference. However, a Case Statement cannot contain a condition that specifies that the statement be executed at the rising or falling of an edge or at the occurrence of an event; as a result, you cannot use the Case Statement to infer a register.
ACTION:	Change the Case Statement to an If Statement, which can contain an edge-triggered or event-triggered condition.

See also:

Implementing Registers
Sections 8.1, 8.7, and 8.8 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Case Statement error at <*location*>: Case Statement choices must cover all possible values of expression

---

CAUSE: In a Case Statement at the specified location in a VHDL Design File (**.vhd**), you specified choices for a Case Statement expression. However, the choices do not cover all possible values of the expression. The choices must cover all possible expression values.

ACTION: Add choices for all possible values of the expression, or add an `OTHERS` choice, which covers all possible values that are not included in the other Case Statement choices.

See also:

Section 8.8 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Case Statement error at <*location*>: Case Statement expression has <*type*> type, but must have discrete type or one-dimensional array type with character type elements

---

CAUSE:	In a Case Statement at the specified location in a VHDL Design File (**.vhd**), you specified a Case Statement expression with the specified type. However, the expression must have a discrete type or a one-dimensional array type that has an element base type that is a character type. This error may occur when using a Case Statement to implement RAM in a design.
ACTION:	Change the Case Statement using temporary variables and nested Case Statements.

See also:

Section 8.8 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Case Statement error at <*location*>: elements of array type Case Statement expression has <*type*> type, but must have character type

---

CAUSE: In a Case Statement at the specified location in a VHDL Design File (**.vhd**), you specified a Case Statement expression with an array type. However, the expression's elements have the specified type, and not a character type. The element base type of an array type Case Statement expression must be a character type. This error may occur when using a Case Statement to implement RAM in a design.

ACTION: Change the Case Statement using temporary variables and nested Case Statements.

See also:

Section 8.8 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Case Statement information at <*location*>: OTHERS choice is never selected

---

CAUSE: In a Case Statement at the specified location in a VHDL Design File (**.vhd**), you specified an `OTHERS` choice. However, the `OTHERS` choice is never selected because the other choices in the Case Statement cover all possible values of the Case Statement expression.

ACTION: To avoid receiving this message in the future, delete the `OTHERS` choice or change the other choices in the Case Statement so they do not cover all the values of the Case Statement expression. Otherwise, no action is required.

See also:

Section 8.8 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Case Statement or Selected Signal Assignment error at <*location*>: OTHERS choice must be last choice in Case Statement or Selected Signal Assignment

---

CAUSE:	In a Case Statement or Selected Signal Assignment at the specified location in a VHDL Design File (**.vhd**), you used the `OTHERS` choice. However, the `OTHERS` choice is not the last choice in the Case Statement or Selected Signal Assignment. The `OTHERS` choice must be the last choice.
ACTION:	Make sure the `OTHERS` choice is last choice in the Case Statement or Selected Signal Assignment.

See also:

Sections 8.8 and 9.5.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Case Statement or Selected Signal Assignment error at <*location*>: OTHERS choice must contain only the OTHERS keyword

---

CAUSE:	In a Case Statement or Selected Signal Assignment at the specified location in a VHDL Design File (**.vhd**), you used the `OTHERS` choice. However, the `OTHERS` choice contains elements. The `OTHERS` choice must contain only the `OTHERS` keyword.
ACTION:	Change the `OTHERS` choice so it contains only the `OTHERS` keyword.

See also:

Sections 8.8 and 9.5.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Case Statement warning at <*location*>: subtype of expression is not locally static

---

CAUSE: In a Case Statement at the specified location in a VHDL Design File (**.vhd**), you specified a Case Statement expression with an unconstrained array subtype. However, the expression's subtype must be a locally static subtype.

ACTION: Change the expression so its subtype is a locally static subtype.

See also:

Sections 7.4 and 8.8 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Component Configuration error at <*location*>: component <*name*> must be associated only to design entities in Binding Indication

---

CAUSE:	In a Component Configuration at the specified location in a VHDL Design File (**.vhd**), you associated the specified component to an item by specifying the item in a Binding Indication for the component. However, the item is not a design entity. You can associate components only to design entities.
ACTION:	Change the Component Configuration so it associates components only to design entities.

See also:

Sections 1.3.2 and 5.2.1of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Component Configuration error at <*location*>: component instance <*name*> in Configuration Specification must specify an instance of component <*name*>

CAUSE: In a Component Configuration at the specified location in a VHDL Design File (**.vhd**), you used a Configuration Specification with the specified component instance and component. However, in a Component Instantiation Statement, you specified a different component for the component instance. A component for a component instance must be the same in a Component Configuration and Component Instantiation Statement.

ACTION: In the Component Configuration, change the component so it is the same as the component in the component instance's Component Instantiation Statement. Or, change the component in the Component Instantiation Statement so it is the same as the component in the Component Configuration.

See also:

Sections 1.3.2, 5.2, and 9.6 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Component Configuration or Component Instantiation Statement error at <*location*>: component <*name*> is used but not declared

---

CAUSE: In a Component Configuration or Component Instantiation Statement at the specified location in a VHDL Design File (**.vhd**), you specified a component. However, you did not declare the component in a Component Declaration. You must declare a component before you can use it.

ACTION: In the Component Configuration or Component Instantiation Statement, use a component that you already declared in a Component Declaration. Or, use a Component Declaration to declare the component that is in the Component Configuration or Component Instantiation Statement. However, do not define a component with the name you used in the Component Configuration or Component Instantiation Statement if you already used the name for a different construct.

See also:

Sections 1.3.2, 4.5, and 9.6 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Component Configuration warning at <*location*>: component <*name*> has already been bound to a design entity -- ignored second Binding Indication

---

CAUSE: In a Component Configuration at the specified location in a VHDL Design File (**.vhd**), you used a Binding Indication to bind the specified component to a design entity. However, you already bound the component to a design entity using a different Binding Indication. As a result, Integrated Synthesis ignored the second Binding Indication.

ACTION: If Integrated Synthesis is using the wrong Binding Indication, or you want to avoid receiving this message in the future, remove one of the Binding Indications from the VHDL Design File. Otherwise, no action is required.

See also:

Sections 1.3.2 and 5.2.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Component Configuration warning at <*location*>: component <*name*> has already been bound to a design entity -- ignored second Binding Indication

---

CAUSE: In a Component Configuration at the specified location in a VHDL Design File, you used the `ALL` keyword and a Binding Indication to bind all instances of a component to a design entity. However, you already bound the specified component to a design entity using a different Binding Indication. As a result, Integrated Synthesis ignored the second Binding Indication for the specified component.

ACTION: If Integrated Synthesis is using the wrong Binding Indication, or you want to avoid receiving this message in the future, remove one of the Binding Indications from the VHDL Design File or change the `ALL` keyword to the `OTHERS` keyword for the Binding Indication. Otherwise, no action is required.

See also:

Sections 1.3.2 and 5.2.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Component Configuration warning at <*location*>: incremental Binding Indications are not supported

---

CAUSE: In a Component Configuration at the specified location in a VHDL Design File (**.vhd**), you used a Binding Indication to bind a component to a design entity. This Binding Indication is an incremental Binding Indication because you already bound the component to the design entity using a different Binding Indication. However, incremental Binding Indications are not supported in the current version of the MAX+PLUS II Advanced Synthesis software.

ACTION: If Integrated Synthesis is using the wrong Binding Indication for the component and design entity, or you want to avoid receiving this message in the future, remove one of the Binding Indications from the VHDL Design File. Otherwise, no action is required.

See also:

Sections 1.3.2 and 5.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Component Configuration warning at <*location*>: treated component <*name*> as 'black box' because Binding Indication associated component to design entity that is not in any defined library

---

CAUSE: In a Component Configuration at the specified location in a VHDL Design File (**.vhd**), you associated the specified component to a design entity by specifying the design entity in a Binding Indication for the component. However, the design entity is not in any of the libraries you specified for the design. As a result, Integrated Synthesis treated the component as a "black box" where the design entity can be specified later, for example, in a separate design that uses this design.

ACTION: To avoid problems in the future processing of the design, make sure the design entity is specified at another time. To avoid receiving this message in the future, specify the design entity in a library you specified and used for the design, or specify and use a new library that contains the design entity. Otherwise, no action is required.

See also:

Section 1.3.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Component Declaration error at <*location*>: port <*name*> is listed for component, but not listed for entity <*name*>

CAUSE: In a Component Declaration at the specified location in a VHDL Design File (**.vhd**), you listed the specified port for a component that is based on the specified entity. However, you did not list the port in the entity's Entity Declaration. The ports you list for a component in a Component Declaration must be the same as the ports you list for the corresponding entity in an Entity Declaration.

ACTION: Make sure the ports for the component in the Component Declaration match the ports in the Entity Declaration for the corresponding entity.

See also:

Sections 1.1 and 4.5 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Component Instantiation Statement error at <*location*>: all generics in component <*name*> must have values

---

CAUSE:	In a Component Instantiation Statement at the specified location in a VHDL Design File (**.vhd**), you instantiated the specified component. However, you did not specify values for all the generics in the component in either the Component Instantiation Statement or the component's Component Declaration. The generics in a component must all have values.
ACTION:	Specify values for all the component's generics in either the Component Instantiation Statement or the Component Declaration.

See also:

Sections 4.5 and 9.6 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Component Instantiation Statement error at <*location*>: argument for conversion function cannot be OPEN keyword

---

CAUSE: In a Component Instantiation Statement at the specified location in a VHDL Design File (**.vhd**), you specified a conversion function for a formal parameter of a component. However, you specified the `OPEN` keyword as the argument for the conversion function. The argument cannot be the `OPEN` keyword; that is, you cannot pass the `OPEN` keyword to a conversion function.

ACTION: Change the argument for the conversion function from the `OPEN` keyword to a valid argument.

See also:

Sections 4.3.2.2 and 9.6 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Component Instantiation Statement error at <*location*>: argument for conversion function cannot contain a formal parameter

---

CAUSE: In a Component Instantiation Statement at the specified location in a VHDL Design File (**.vhd**), you specified a conversion function for a formal parameter of a component. However, you specified both a formal parameter and an actual parameter as the argument for the conversion function. The argument cannot contain a formal parameter. The argument must contain only an actual parameter; that is, you must pass only an actual parameter to a conversion function.

ACTION: Remove the formal parameter from the argument for the conversion function.

See also:

Sections 4.3.2.2 and 9.6 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Component Instantiation Statement error at <*location*>: conversion function for formal parameter must contain only one argument

---

CAUSE: In a Component Instantiation Statement at the specified location in a VHDL Design File (**.vhd**), you specified a conversion function for a formal parameter of a component. However, you specified two or more arguments for the conversion function. The conversion function must contain only one argument; that is, you must pass only one argument to a conversion function.

ACTION: Change the conversion function so that it uses only one argument.

See also:

Sections 4.3.2.2 and 9.6 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Conditional Signal Assignment error at <*location*>: conditional waveforms must have same number of elements

---

CAUSE: In a Conditional Signal Assignment at the specified location in a VHDL Design File (**.vhd**), you used more than one conditional waveform. However, the different conditional waveforms do not contain the same number of elements. For example, the Conditional Signal Assignment in the following code contains one conditional waveform with one element, and one conditional waveform with two elements.

```
ENTITY my_entity IS
   PORT
   (
      i: IN BIT
   );
END my_entity;

ARCHITECTURE a OF my_entity IS
   SIGNAL s1: bit_vector(1 downto 0);
BEGIN
   s1 <= "0" WHEN i = '1' ELSE
         "11";
END a;
```

All the conditional waveforms in a Conditional Signal Assignment must have the same number of elements.

ACTION: Change the conditional waveforms so they all have the same number of elements. For the example above, you can change the first conditional waveform so it has two elements.

See also:

Sections 9.5.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Conditional Signal Assignment error at <*location*>: waveform that contains UNAFFECTED keyword cannot contain any waveform elements

---

CAUSE:	In a Conditional Signal Assignment at the specified location in a VHDL Design File (**.vhd**), you used the `UNAFFECTED` keyword for a waveform. However, you also used one or more waveform elements in the waveform. A waveform that contains the `UNAFFECTED` keyword cannot contain any waveform elements.
ACTION:	Remove either the `UNAFFECTED` keyword or the waveform element(s) from the waveform.

See also:

Sections 8.4 and 9.5.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Configuration Declaration error at <*location*>: architecture <*name*> must be declared for entity <*name*>

CAUSE: In a Configuration Declaration at the specified location in a VHDL Design File (**.vhd**), you bound the specified architecture to the specified entity. However, you did not declare the architecture for the entity in an Architecture Body. You must declare an architecture for an entity before you can bind the architecture to the entity.

ACTION: Make sure the architecture you bound to the entity is declared for the entity in an Architecture Body, and that you included the VHDL Design File that contains the Architecture Body in the project.

See also:

Sections 1.2 and 1.3 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Configuration Declaration error at <*location*>: entity must be bound to one declared architecture in a Configuration Declaration -- ignored architecture <*name*>

CAUSE: In a Configuration Declaration at the specified location in a VHDL Design File (**.vhd**), you bound the specified architecture to the specified entity for simulation. However, you did not declare the architecture for the entity in an Architecture Body, or the architecture is not visible. As a result, Integrated Synthesis ignored the architecture.

ACTION: Make sure the architecture you bound to the entity is declared for the entity in an Architecture Body, and that you included the VHDL Design File that contains the Architecture Body in the project. Also, make sure the architecture is visible.

See also:

Sections 1.2 and 1.3 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Constant Declaration error at <*location*>: constant <*name*> must have initial value

---

CAUSE: In a Constant Declaration at the specified location in a VHDL Design File (**.vhd**), you declared the specified constant. However, you did not specify an initial value for the constant. The constant must have an initial value.

ACTION: Specify an initial value for the constant.

See also:

Section 4.3.1.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Constant Declaration error at <*location*>: constant cannot be file type or access type

---

CAUSE: In a Constant Declaration at the specified location in a VHDL Design File (**.vhd**), you declared a constant with a file type or an access type. However, a constant cannot have a file type or an access type.

ACTION: Change the type of the constant so that it is neither a file type nor an access type.

See also:

Section 4.3.1.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Constant Declaration error at <*location*>: initial value for constant must be a constant

---

CAUSE: In a Constant Declaration at the specified location in a VHDL Design File (**.vhd**), you assigned an initial value to the constant you are declaring. However, the initial value is not a constant; for example, you may have used a variable as the initial value. The initial value for a constant you declare in a Constant Declaration must be a constant.

ACTION: Change the initial value so it is a constant, or, if you want the value of the declared constant to change, use a Variable Declaration to replace the declared constant with a variable..

See also:

Section 4.3.1.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Default Binding Indication error at <*location*>: actual generic <*name*> in Generic Map Aspect must have same type as formal generic with same name in entity <*name*>

CAUSE:	In a Default Binding Indication at the specified location in a VHDL Design File (**.vhd**), you specified a Generic Map Aspect that associates actual generics with the formal generics in the specified entity. However, the specified actual generic does not have the same type as the entity's formal generic with the same name. The actual generic and its corresponding formal generic must have the same type.
ACTION:	Change the type of the actual generic so it is the same as the type of the corresponding formal generic in the entity.

See also:

Sections 5.2.1.2 and 5.2.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Default Binding Indication error at <*location*>: formal generic <*name*> in named Association List in Generic Map Aspect must have corresponding formal generic in entity <*name*>

CAUSE:	In a Default Binding Indication at the specified location in a VHDL Design File (**.vhd**), you used a named Association List in a Generic Map Aspect to associate actual generics with the formal generics in the specified entity. However, the entity does not have a formal generic with the same name as the specified formal generic in the Generic Map Aspect. A formal generic in a Generic Map Aspect of a Default Binding Indication must have a corresponding formal generic in the associated entity.
ACTION:	Make sure the Generic Map Aspect only associates actual generics to formal generics that exist in the associated entitiy.

See also:

Sections 3.2.2, 5.2.1.2, 5.2.2 9.1, and 12.2.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Entity Declaration error at <*location*>: ports must be constrained

---

CAUSE: In an Entity Declaration at the specified location in a VHDL Design File (**.vhd**), you specified an unconstrained port. However, the ports in an Entity Declaration must be constrained.

ACTION: Make sure all the ports in the Entity Declaration are constrained by specifying sizes or ranges for the ports.

See also:

Section 1.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Entity Declaration warning at <*location*>: primary unit already exists -- overwriting existing primary unit with new primary unit

---

CAUSE:	In an Entity Declaration at the specified location in a VHDL Design File (**.vhd**), you declared a primary unit for a physical type. However, you already declared the primary unit in a different Entity Declaration for the same physical type. As a result, Integrated Synthesis replaced the existing primary unit with the new primary unit.
ACTION:	If you do not want Integrated Synthesis to overwrite the existing primary unit, or to avoid receiving this message in the future, change the new primary unit so it has a unique name.

See also:

Section 1.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: <*BUS or REGISTER*> signal kind must be in Signal Declaration or Interface Signal Declaration

---

CAUSE:	In an object declaration at the specified location in a VHDL Design File (**.vhd**), you used the `BUS` or `REGISTER` signal kind. However, the object declaration is not a Signal Declaration or Interface Signal Declaration. You can use the `BUS` or `REGISTER` signal kind only in Signal Declarations or Interface Signal Declarations.
ACTION:	Change the object declaration to a Signal Declaration or Interface Signal Declaration, or delete the signal kind.

See also:

Sections 4.3.1.2 and 4.3.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: <*name*> type is used but not declared

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the specified type. However, you did not declare the type in a Type Declaration, and the type is not a predefined type. To use a type, you must define the type or the type must be a predefined type.

ACTION: Replace the type with a previously defined or predefined type, or define the type in a Type Declaration.

See also:

Sections 3, 4.1, and 14 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: <*type*> type is used but not declared as an array type

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the specified type as if it were an array type. However, the type is not an array type. For example, the Signal Declaration in the following code specifies the range of `(0 to 1)` for the `STD_LOGIC` type, but the `STD_LOGIC` type is not an array type:

```
ENTITY example IS
   PORT ( ex1:  OUT  BIT );
END ENTITY;

ARCHITECTURE error_test OF example IS
   SIGNAL sig3: STD_LOGIC(0 to 1);

BEGIN
   ex1 <= '1';
END error_test;
```

ACTION: Change the type to an array type, or make sure you do not use the type as an array type. In the example above, `STD_LOGIC` can be replaced with `STD_LOGIC_VECTOR`.

See also:

Sections 3.2.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: access type <*name*> is assigned to object, but type of elements in access type must be the same as <*name*> type of object

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you assigned the specified access type to an object. However, the type of the elements in the access type is not the same as the specified type of the object. The type of the access type's elements and the type of the object must be the same.

ACTION: Make sure the element type of the access type matches the object type.

See also:

Section 3.3 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: actual for formal port <*name*> of mode <*text*> cannot be expression

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used an expression as an actual for the specified formal port. However, the formal port has the specified mode, which is not the mode `IN`. You can use an expression as an actual for a formal port only if the formal port has a mode of `IN`.

ACTION: Change the actual to a port or signal, or change the mode of the formal port to `IN`.

See also:

Sections 1.1.1.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: actual port <*name*> of mode <*text*> cannot be associated with formal port <*name*> of mode <*text*>

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you associated the specified actual port with the specified formal port. However, you cannot associate an actual port of the specified mode with a formal port of the specified mode. For example, you cannot associate an actual port of mode `OUT` with a formal port of mode `IN`.

ACTION: Make sure the modes of the actual port and the formal port are compatible.

See also:

Sections 1.1.1.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: all elements in object alias must be assigned values

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you assigned value(s) to an index or slice of an object alias of array type. However, you must assign values to the entire object alias, that is, to all elements in the object alias.

ACTION: Assign values to all elements in an object alias.

See also:

Section 4.3.3.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: all elements of record type object cannot be accessed with suffix .all

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you attempted to access all the elements in a record type object using the format <*record type object name*>`.all`. However, you cannot use the suffix `.all` with a record type object. For example, in the Signal Assignment Statement in the following code, you cannot use `exa.all` and `exb.all` to access all the elements in signals `exa` and `exb` because both signals have a record type.

```
TYPE example IS
   RECORD
      ex1 : INTEGER range 1 to 5;
      ex2 : INTEGER range 1 to 6;
      ex3 : INTEGER range 1 to 7;
   END example;
SIGNAL exa : example;
SIGNAL exb : example;

exa.all <= exb.all;
```

ACTION: Remove the `.all` suffix from the record type object name. You can also replace a record type object with an aggregate that specifies all the elements in the object. In the example above, you can use either of the following Signal Assignment Statements:

```
exa <= (ex1 => exb.ex1, ex2 => exb.ex2, ex3 => exb.ex3);
exa <= exb;
```

See also:

Sections 3.2.2 and 7.3.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: allocator with NEW keyword used for <*type*> type, but must be used for access type

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used an allocator for the specified type (that is, you used the `NEW` keyword with the type). However, you can use an allocator only for an access type.

ACTION: Make sure that you use allocators only for access types.

See also:

Sections 3.3 and 7.3.6 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: allocators are not synthesizable

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used an allocator to create an object by allocating free memory. However, allocators are not synthesizable.

ACTION: Make sure you do not use any allocators in the design, that is, you do not try to create any objects by allocating free memory.

See also:

Section 7.3.6 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: attribute <*name*> cannot be used in constraint

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the specified attribute in a constraint. However, the attribute cannot be used in constraints. Only certain attributessuch as `'BASE`, `'RANGE`, and `'REVERSE_RANGE` can be used to return constraints from a name.

ACTION: Remove the attribute from the constraint.

See also:

Section 14 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: Can't access enumeration literal to the left of enumeration literal <*name*> in enumeration type <*name*>

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you attempted to access an enumeration literal to the left of the specified enumeration literal in the specified enumeration type. However, the specified enumeration literal is the first enumeration literal in the enumeration type, and therefore it has no enumeration literals to its left for Integrated Synthesis to access. For example, the `'PRED` predefined attribute in the following code attempts to access an enumeration literal that is positioned to the left of enumeration literal `st0` in the enumeration type `StateType`. However, `st0` is the first enumeration literal in `StateType`.

```
TYPE StateType IS (st0, st1, st2);
SIGNAL current_state: StateType;
SIGNAL next_state: StateType;

PROCESS(i, current_state)
BEGIN
   current_state <= st2;
   CASE current_state is
      WHEN st2 =>
         next_state <= StateType'PRED(st0);
      WHEN others =>
         next_state <= st2;
   END CASE;
END PROCESS;
```

ACTION: Make sure you do not try to access an enumeration literal that is positioned to the left of the first enumeration literal in an enumeration type.

See also:

Section 3.1.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: can't determine definition of name because multiple possible definitions exist

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used a name that has more than one possible definition. As a result, Integrated Synthesis cannot determine the definition of the name.

ACTION: Make sure the design clearly specifies only one definition for the name. If necessary, change the current name to a new, unique name.

See also:

Section 6 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: can't determine definition of operator <*text*> -- found <*number*> possible definitions

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the specified operator. However, Integrated Synthesis cannot determine the definition of the operator because no definition exists or Integrated Synthesis found the specified number of different possible definitions for the operator.

ACTION: Make sure the design clearly specifies the definition of the operator.

See also:

Section 7.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: can't determine object and type associated with indexed name or signature name near text <*text*> -- found <*number*> possible objects and types

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used an indexed name or a signature name near the specified text. Integrated Synthesis cannot determine the object and type associated with the indexed or signature name because it found the specified number of different possible objects and types for the indexed or signature name, or the indexed or signature name is not visible.

ACTION: Make sure the design clearly specifies the object and type associated with the indexed or signature name.

See also:

Sections 2.3.2, 4.3.3, 5.1, 6.4, 6.6 and 10.3 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: can't determine object and type associated with selected name near text <*text*> -- found <*number*> possible objects and types

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used a selected name near the specified text. Integrated Synthesis cannot determine the object and type associated with the selected name because it found the specified number of different possible objects and types for the selected name, or the selected name is not visible.

ACTION: Make sure the design clearly specifies the object and type associated with the selected name. For example, you can explicitly specify an object and type for a selected name using a Use Clause.

See also:

Sections 6.3, 10.3, and 10.4 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: can't determine type of object at or near bit string literal <*text*> -- found <*number*> possible types

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used an object at or near the specified bit string literal. Integrated Synthesis cannot determine the type of the object because it found the specified number of different possible types for the object, or it found no possible object type. This error may occur because you are using the IEEE Std 1076-1987 version of VHDL in the VHDL Design File.

ACTION: Make sure the design clearly specifies the type of the object. For example, you can explicitly specify a type for an object using a qualified expression, or you can make sure an Architecture Body for an entity uses a declared entity.

See also:

Sections 1.2, 7.3.4, 10.3, and 13.7 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: can't determine type of object at or near character <*character*> -- found <*number*> possible types

---

CAUSE:	In a VHDL Design File (**.vhd**) at the specified location, you used an object at or near the specified character. Integrated Synthesis cannot determine the type of the object because it found the specified number of different possible types for the object.
ACTION:	Make sure the design clearly specifies the type of the object. For example, you can explicitly specify a type for an object using a qualified expression, or make sure that an Architecture Body for an entity uses a declared entity.

See also:

Sections 1.2, 7.3.4, and 10.3 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: can't determine type of object at or near identifier <*name*> -- found <*number*> possible types

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used an object at or near the specified identifier. Integrated Synthesis cannot determine the type of the object because it found the specified number of different possible types for the object, or it found no possible object type. For example, Integrated Synthesis cannot determine the type of object `i` or object o because two Type Declarations declare two different types named `color`:

```
PACKAGE p1 IS
   TYPE color IS (red, green, blue);
END p1;

PACKAGE p2 IS
   TYPE color IS (black, white);
END p2;

USE work.p1.all;
USE work.p2.all;

ENTITY my_entity IS
   PORT
   (
      i : IN    color;
      o : OUT   color;
   );
END my_entity;

ARCHITECTURE a OF my_entity IS
BEGIN
   o <= i;
END a;
```

ACTION: Make sure the design clearly specifies the type of the object. For example, you can explicitly specify a type for an object using a qualified expression, or you can make sure an Entity Declaration or Architecture Body uses an unambiguous Type Declaration. For the example above, you can change the `color` type to two separate types named `color1` and `color2`.

See also:

Sections 1.1, 1.2, 4.1, 7.3.4, 10.3, and 13.3 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: can't determine type of object at or near string <*text*> -- found <*number*> possible types

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used an object at or near the specified string. Integrated Synthesis cannot determine the type of the object because it found the specified number of different possible types for the object, or it found no possible object type.

ACTION: Make sure the design clearly specifies the type of the object. For example, you can explicitly specify a type for an object using a qualified expression, or you can make sure an Architecture Body for an entity uses a declared entity.

See also:

Sections 1.2, 7.3.4, 10.3, and 13.6 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: can't execute unsupported pragma for function <*name*>

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used a pragma for the specified function. However, the MAX+PLUS II Advanced Synthesis software does not support the pragma, and therefore cannot execute the pragma.

ACTION: Replace the unsupported pragma with a different, supported pragma.

VHDL error at <*location*>: can't read or update value of interface object <*name*> of mode LINKAGE

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used an interface object of mode `LINKAGE`. However, Integrated Synthesis cannot read or update the value of the interface object because you did not use the interface object as an actual for a formal interface object of mode `LINKAGE`.

ACTION: Make sure the interface object is an actual for a formal interface object, and that the mode of both interface objects is `LINKAGE`.

See also:

Sections 4.3.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: can't read value of interface object <*name*> of mode OUT

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you assigned an interface object of mode `OUT` to an object. However, Integrated Synthesis cannot read the value of an interface object of mode `OUT` that you assign to an object.

ACTION: Change the mode of the interface object to `IN`, `INOUT`, `BUFFER`, or `LINKAGE`. Make sure that all other requirements for reading a value of an interface object of mode `IN`, `INOUT`, `BUFFER`, or `LINKAGE` are met.

See also:

Sections 4.3.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: can't synthesize magnitude comparator for user-defined types

---

CAUSE:	In a VHDL Design File (**.vhd**) at the specified location, you used a magnitude comparator for a type that you defined. However, Integrated Synthesis cannot synthesize magnitude comparators for user-defined types.
ACTION:	Use explicitly defined constants with the magnitude comparator.

See also:

Sections 3 and 7.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: can't update value of interface object <*name*> of mode IN

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used an interface object of mode `IN`. However, Integrated Synthesis cannot update the value of an interface object of mode `IN`.

ACTION: Change the mode of the interface object to `OUT`, `INOUT`, `BUFFER`, or `LINKAGE`. Make sure that all other requirements for updating the value of an interface object of mode `OUT`, `INOUT`, `BUFFER`, or `LINKAGE` are met.

See also:

Sections 4.3.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: can't write to interface object <*name*> of mode IN

CAUSE: At the specified location in a VHDL Design File (**.vhd**), you used an interface object of mode `IN`. However, Integrated Synthesis cannot write to the value of an interface object of mode `IN`.

ACTION: Change the mode of the interface object to `OUT` or `INOUT`, or use an existing interface object of mode `OUT` or `INOUT`.

See also:

Section 4.3.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: character <*name*> used but not declared for type <*name*>

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the specified character for the specified type. However, you did not declare the character for the type in a Type Declaration. You must declare the character before you can use it for a type.

ACTION: Declare the character for the type, or use a character that is already declared for the type.

See also:

Sections 3 and 4.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: construct <*name*> is used but not defined as a record element

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location (for example, in a record aggregate), you used the specified construct as a record element. However, you did not define the construct as a record element in a record Type Declaration.

ACTION: Replace the construct with a valid record element.

See also:

Sections 3.2.2 and 7.3.2.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: digit <*character*> in based literal <*name*> must be smaller or equal to base

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you specified a based literal that contains a digit that is larger than the based literal base. The digit must be smaller than or equal to the base.

ACTION: Make sure the based literal contains a digit that is smaller than or equal to the based literal base.

See also:

Section 13.4.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: discrete range has <*type*> type, but must have discrete type

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used a discrete range. However, the discrete range has the specified type. The discrete range must have a discrete type, for example, an integer type or an enumeration type.

ACTION: Change the type of the discrete range to a discrete type.

See also:

Sections 3.2.1, 4.1, and 8.9 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: elements in exponent operations must be constant

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used non-constant elements in an exponent operation. However, an exponent operation's elements must be constant.

ACTION: Change the exponent operation's elements so they are constant.

See also:

Section 13 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: entity <*name*> is used but not declared

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the specified entity. However, you did not declare the entity in an Entity Declaration. You must declare an entity before you can use it.

ACTION: Use an Entity Declaration to declare the entity, or remove the reference to the entity from the design.

See also:

Section 1.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: entity <*name*> is used in Architecture Body, Configuration Specification, or Component Configuration, but is not defined

---

CAUSE: In an Architecture Body, Configuration Specification, or Component Configuration at the specified location in a VHDL Design File (**.vhd**), you used the specified entity. However, you did not declare the entity in an Entity Declaration. You must define the entity before you can use it.

ACTION: Replace the entity with a previously defined entity, or define the entity in an Entity Declaration. However, do not define an entity with the name you used in the Architecture Body, Configuration Specification, or Component Configuration if you already used the name for a different construct.

See also:

Sections 1.1, 1.2, 1.3, and 5.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: entity <*name*> must be in current project's work library

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the specified entity, which is not in the **work** library of the current project. All entities must be in the current project's **work** library.

ACTION: Remove the entity from the VHDL Design File, or add the entity to the current project's **work** library.

See also:

Sections 11.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: entity aspect <*name*> must identify an entity or configuration

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used an entity aspect that does not identify an entity or configuration. However, an entity aspect must identify an entity or configuration.

ACTION: Change the entity aspect so it identifies an entity or configuration.

See also:

Sections 5.2.1.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: entity must be in current project's work library

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used an entity that is not in the **work** library of the current project. All entities must be in the current project's **work** library.

ACTION: Remove the entity from the VHDL Design File, or add the entity to the current project's **work** library.

See also:

Sections 11.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: Exit Statement must be used inside Loop Statement

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used an Exit Statement outside of a Loop Statement. You can use an Exit Statement only inside of a Loop Statement.

ACTION: Make sure the design uses Exit Statements only within Loop Statements.

See also:

Sections 8.9 and 8.11 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: exponentiation with negative exponent must be used only for real type objects

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used an exponentiation with a negative exponent for an object that does not have a real type. You can use an exponentiation with a negative exponent only for a real type object.

ACTION: Make sure the design contains exponentiation with negative exponents only for real type objects.

See also:

Section 3.1.4.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: formal parameter <*name*> must have actual or default value

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the specified formal parameter. However, you did not assign an actual value to the parameter, and the parameter does not have a default value. For example, in the following code, the function `exa` has the parameter `r`, but the Signal Assignment Statement specifies two values for the parameter `s` and no value for `r`.

```
ENTITY example IS
   PORT
   (
      a : IN   BIT;
      b : IN   BIT;
      o : OUT  BIT
   );
END example;
ARCHITECTURE a OF example IS
   FUNCTION exa (s,r : BIT) RETURN BIT IS
   BEGIN
      RETURN (s and r);
   END exa;
BEGIN
   o <= exa (s=>a, s=>b);
END a;
```

The parameter must have a value. In the example above, the parameter `r` must have a value.

ACTION: Specify an actual value for the parameter, or make sure the parameter has a default value.

See also:

Sections 2.2 and 4.3.2.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: formal parameter <*name*> is already associated

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, the named formal parameter is already associated with an actual parameter. A likely reason for this is that the formal is a constant generic or inputparameter to which you are now assigning a full value.

ACTION: Check your parameter lists for the above possibilities, and correct the statement.

See also:

The IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: formal parameter must have one argument

---

CAUSE:	In a VHDL Design File (**.vhd**) at the specified location, you used a formal parameter that does not have any arguments. However, the formal parameter must have one argument.
ACTION:	In the formal parameter, add one argument based on where the parameter is used.

See also:

The IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: function or entity <*name*> is declared, but is a homograph of a function or entity previously declared in same declarative region

---

CAUSE:	In a VHDL Design File (**.vhd**) at the specified location, you declared the specified function or entity, but you already declared the function or entity with the same identifier in the same declarative region. You can declare a function or entity only once in a declarative region.
ACTION:	Remove one of the declarations, or change the identifier in one of the declarations so the two declarations are different.

See also:

Section 10.3 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: generic <*name*> cannot be used in its own interface list

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you defined a generic in an interface list. However, you used the generic to define itself. For example, the interface list for the `add_width` generic in the following code defines the `add_width` generic using the `add_width` generic:

```
GENERIC (add_width: integer := add_width);
```

Trying to define a generic using the same generic causes a circular dependency.

ACTION: Change the interface list so it does not use the generic to define itself.

See also:

Sections 1.1.1.1 and 4.3.2.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: generic <*name*> does not exist on binding unit

CAUSE: The named generic does not exist in the binding unit it is specified for. For example, in the following code, though `size` is a properly formed generic, the generic `example` is specified in the binding component, but is not defined in the entity `rreg`:

```
ENTITY rreg IS
   GENERIC (size : INTEGER := 3);
   PORT
      (          clk : IN  STD_LOGIC;
         d    : IN  STD_LOGIC_VECTOR(size-1 DOWNTO 0);
         q    : OUT STD_LOGIC_VECTOR(size-1 DOWNTO 0)      );
END rreg;

ARCHITECTURE a OF rreg IS
BEGIN
   p1: PROCESS (clk)
   BEGIN
      q <= d;
   END PROCESS;
END a;


ENTITY top IS
   PORT
      (          clk : IN     STD_LOGIC;
          d    : IN     STD_LOGIC_VECTOR(1 DOWNTO 0);
          q    : BUFFER STD_LOGIC_VECTOR(1 DOWNTO 0)      );
END top;

ARCHITECTURE b OF top IS
   COMPONENT rreg IS
      GENERIC (size : INTEGER; example : INTEGER);
      PORT
         (             clk : IN  STD_LOGIC;
             d    : IN  STD_LOGIC_VECTOR(size-1 DOWNTO 0);
             q    : OUT STD_LOGIC_VECTOR(size-1 DOWNTO 0)         );
   END COMPONENT;

BEGIN
   E1: rreg
      GENERIC MAP(size => 2)
      PORT MAP (d => d, clk => clk, q => q);
END b;
```

ACTION: Remove the offending generic specification, or modify the Entity Declaration to include the proper definition of the generic.

See also:

Sections 1.1 and 5.2.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: ignored construct <*name*> because of previous errors

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the specified construct. However, because of other errors that occurred in this construct, Integrated Synthesis ignored the construct.

ACTION: Fix the other errors in the construct.

VHDL error at <*location*>: illegal <> in expression

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the undefined range symbol (<>) in an expression. However, the expression must have a defined range. For example, the expression may not be a type, or may be a subtype of a constrained type.

ACTION: Change the undefined range symbol in the expression to a defined range, or change the expression so it can contain the undefined range symbol.

See also:

Sections 3.2.1 and 7 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: illegal formal in expression

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used a formal parameter in an expression. However, the formal parameter is illegal for this expression. A common cause of this error is when a discrete range is being inferred from the type of the formal parameter. Refer to Section 6 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual* for more information on the restrictions for formal parameters.

ACTION: Remove the formal parameter, or make sure the formal parameter is legal.

See also:

Sections 2.1 and 7 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: illegal formal parameter

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used a formal parameter. However, the formal parameter is illegal, for example, the implied type of the formal parameter may not be the same as the type of the return value in a Subprogram Declaration. Refer to Section 6 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual* for more information on the restrictions for formal parameters.

ACTION: Remove the formal parameter, or make sure the formal parameter is legal.

See also:

Section 2.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: illegal Function Call in assignment

---

CAUSE: In an assignment at the specified location in a VHDL Design File (**.vhd**), you used a Function Call or Type Conversion. However, the Function Call or Type Conversion is illegal in the assignment, for example, a Type Conversion function or the Type Conversion of a formal parameter may be missing a type or contain an incorrect type.

ACTION: Remove the Function Call or Type Conversion, or make sure the Function Call or Type Conversion is legal.

See also:

Sections 7.3.3 and 7.3.5 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: illegal index constraint in expression

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used an index constraint in an expression. However, the index constraint is illegal for the expression, for example, you may not be able to use an index constraint for a type in the expression, or you may have incorrectly specified the index constraint.

ACTION: Remove the index constraint, or make sure the index constraint is legal.

See also:

Section 7 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: illegal name for assignment

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used a name in an assignment. However, the name is illegal for the assignment, for example, the name may be in an array or indexed name but is not an indexed name, or the name may have an incorrect type.

ACTION: Remove the name, or make sure the name is legal.

See also:

Sections 2.4, 3, and 6 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: illegal name on sensitivity list

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used a name in a sensitivity list. However, the name is illegal for a sensitivity list.

ACTION: Remove the name, or make sure the name is legal.

See also:

Sections 6.1, 8.1, and 9.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: illegal named association in array index

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used a named association in an array index. However, the named association is illegal for the array index, for example, the named association may have an incorrect type.

ACTION: Remove the named association, or make sure the named association is legal.

See also:

Section 6 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: illegal named association in index constraint

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used a named association in an index constraint. However, the named association is illegal for the index constraint, for example, the named association may have an incorrect type.

ACTION: Remove the named association, or make sure the named association is legal.

See also:

Section 6.4 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: illegal named association in type conversion

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used a named association in a type conversion. However, the named association is illegal for the type conversion.

ACTION: Remove the named association, or make sure the named association is legal.

See also:

Section 6 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: illegal range constraint in expression

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used a range constraint in an expression. However, the expression cannot contain a range constraint, or cannot contain a range constraint in the context in which the expression is being used. For example, you may have used a range constraint for an integer type.

ACTION: Remove the range constraint, or change the expression so it can contain a range constraint.

See also:

Sections 3.1 and 7 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: illegal range in expression

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used a range in an expression. However, the expression cannot contain a range, or cannot contain a range in the context in which the expression is being used. For example, you may have used a range in a value that you assigned to an integer type object.

ACTION: Remove the range, or change the expression so it can contain a range.

See also:

Section 7 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: incorrect number of elements in target aggregate

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, the number of elements in the target aggregate differs from the number needed for the assignment.

ACTION: Modify the statement to include the correct number of elements

See also:

Sections 7.3.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: index constraint is not compatible with range

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, a constraint cannot be applied to the range because they are not compatible. For example, in the following code, BIT_VECTOR is defined over NATURAL, so a subtype cannot be defined over INTEGER:

```
type BIT_VECTOR is ARRAY (NATURAL RANGE <>) of BIT;
subtype MY_BIT_VECTOR is BIT_VECTOR (INTEGER RANGE <>);
```

ACTION: Correct the restriction to lie within the bounds of the base type.

VHDL error at <*location*>: index of array type that is associated with the 0, 1, or Z enumerated value must belong to range of array subtype

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used an array type index that is associated with the `0`, `1`, or `Z` enumerated value for the array subtype. However, the index is not in the range you declared for the array subtype in a Type Declaration. The index must be in the array subtype range.

ACTION: Change the index value or the array subtype range so the index associated with the enumerated value belongs to the array subtype range.

See also:

Sections 3.2.1, 4.1, 6.4, and 6.5 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: index of object of array type <*type*> must have <*number*> dimensions

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you specified an index of an object with the specified array type. However, the index does not have the same number of dimensions as you specified for the object in the object type's Type Declaration. The index of an object must have the same number of dimensions as the object. The text of the message indicates the number of dimensions the index must have.

ACTION: Change the object index or the object so they have the same number of dimensions.

See also:

Sections 3.2.1, 4.1, and 6.5 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: index of object of array type <*type*> must have <*number*> dimensions

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you specified an index of an object with the specified array type. However, the index does not have the same number of dimensions as you specified for the object in the object type's Type Declaration. The index of an object must have the same number of dimensions as the object. The text of the message indicates the number of dimensions the index must have.

ACTION: Change the object index or the object so they have the same number of dimensions.

See also:

Sections 3.2.1, 4.1, and 6.5 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: indexed object is assigned to target object, but object index must belong to object index range

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you assigned an index of an array type object to a target object. However, the index you specified for the object is not within the bounds of the index range that you specified for the object. For example, the Signal Assignment Statement in the following code assigns the indexed object `i(4)` to a target object, but the index `4` does not belong to the index range of object `i`, which is `(0 to 3)`:

```
ENTITY example IS
   PORT
   (
      clk      : IN     std_logic;
      i        : IN     std_logic_vector (0 to 3);
      o        : OUT    std_logic_vector (0 to 3)
   );
END example;
ARCHITECTURE ex1 OF example IS
BEGIN
   o(3) <= i(4);
END ex1;
```

The index of the object must belong to the index range you specified for the object.

ACTION: Change the index for the object, or change the bounds of the object index range to include the index. In the example above, you must change `i(4)` to `i(0)`, `i(1)`, `i(2)`, or `i(3)`; or change the index range `(0 to 3)` to include the index `4`.

See also:

Sections 3.2.1, 4.3.1, 6.4, 8.4, and 8.5 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: integer literal cannot have negative exponent

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used an integer literal with a negative exponent. However, an integer literal cannot have a negative exponent.

ACTION: Remove the negative exponent, or change the exponent so it is not a negative value.

See also:

Section 3.1.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: item <*name*> cannot be used as subprogram

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the specified item as a subprogram, that is, as a function or procedure. However, the item cannot be used as a subprogram.

ACTION: Replace the item with a valid subprogram.

See also:

Section 2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: item <*name*> must be a correctly specified range

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the specified item as a range. However, you did not correctly specify the item as a range.

ACTION: Correct the range in terms of the restrictions described in the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*.

See also:

Section 3.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: item cannot be assigned value

CAUSE: In an assignment at the specified location in a VHDL Design File (**.vhd**), you assigned a value to an item. However, you cannot assign values to the item. For example, in the following code, a Signal Assignment Statement assigns the value `i` to `o'delayed`, which is an attribute and therefore cannot be assigned a value:

```
ENTITY example IS

   PORT
   (
      i   :   IN   BIT;
      o   :   OUT  BIT
   );

END example;


ARCHITECTURE ex1 OF example IS
BEGIN
   PROCESS(i)
   BEGIN
      o'delayed <= i;
   END PROCESS;
END ex1;
```

ACTION: Remove the assignment to the item.

See also:

Sections 8.4 and 8.5 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: left bound of range must be a constant

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you specified a range. However, the left bound of the range you specified is not a constant. The left bound of a range must be a constant.

ACTION: Change the left bound of the range to a constant.

See also:

Section 3.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: left bound of slice must belong to range of corresponding object

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you specified a slice of an object that has an array type. However, the left bound of the range you specified for the slice exceeds the range of the array type object. The left bound of the slice must belong to the object range.

ACTION: Change the left bound of the slice so the bound belongs to the range of the object.

See also:

Section 6.5 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: name <*name*> cannot be used because it is already used for a previously declared item

---

CAUSE: At the specified location in a VHDL Design File (**.vhd**), you declared the specified name for an item (for example, for an object or type). However, you cannot use the name for the item because you already used the name for another item. You cannot use a name for more than one item.

ACTION: Use a different name for the item you are declaring.

See also:

Sections 6 and 10.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: name <*name*> must represent signal

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the specified name. However, the name must represent a signal.

ACTION: Make sure the name represents a signal.

See also:

Sections 4.3.1.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: name <*name*> used at end of construct must match name specified at beginning of construct

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the specified name at the end of a construct. However, the name does not match the name you previously specified for the construct. For example, the Architecture Body in the following code contains the name `ex2` after the `END` keyword, but the name of the Architecture Body is specified as `ex1`:

```
ARCHITECTURE ex1 OF example IS
...
END ex2;
```

You must use the same name throughout the construct.

ACTION: Make sure you use the same name throughout the construct. In the example above, you must change the name after the `END` keyword to `ex1`, or change the name you specified for the Architecture Body to `ex2`.

See also:

Section 6 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: name cannot be assigned value

CAUSE: In an assignment at the specified location in a VHDL Design File (**.vhd**), you assigned a value to a name. However, you cannot assign a value to the name. For example, you may have assigned a value to a variable with an access type.

ACTION: Remove the assignment to the name.

See also:

Sections 8.4 and 8.5 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: name in assignment contains illegal prefix

CAUSE: In an assignment in a VHDL Design File (**.vhd**) at the specified location, you used a prefix in a name. However, the prefix cannot be used as a prefix because of its type. For example, in the following code, the signal `s1` does not have a record type and therefore `s1` cannot be use as a prefix in the name `s1.ex1`:

```
ENTITY example IS
   PORT
   (
      i  : IN   BIT;
      o  : OUT  BIT
   );

END example;
ARCHITECTURE exa OF example IS
   TYPE ex IS (ex1, ex2);
   SIGNAL s1, s2 : ex;
BEGIN
   PROCESS (i)
   BEGIN

      ...

      IF s1.ex1 > s2 THEN
         o <= '1';
      ELSE
         o <= '0';
      END IF;
   END PROCESS;
END exa;
```

ACTION: Make sure the prefix is legal for the name. If necessary, remove the prefix from the name.

See also:

Section 3 and 6.3 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: NOT unary operator cannot be applied to enumeration value <*text*> for enumeration type <*type*> because value contains non-bit elements

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you applied the `NOT` unary operator to the specified enumeration value for the specified enumeration type. However, the value for the enumeration type contains elements that do not have a bit subtype. You cannot apply a `NOT` unary operator to an enumeration value that contains non-bit elements.

ACTION: Make sure you apply the `NOT` unary operator only to enumeration values that contain only elements that have a bit subtype.

See also:

Section 7.2.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: object <*name*> is used but not declared

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the specified object. However, you did not declare the object. This error can occur because you did not specify a declaration for the object, you used an incorrect object name in a declaration, or you placed the declaration in an incorrect position (for example, you may have placed the Subprogram Declaration for a subprogram after the call to the subprogram in the VHDL Design File). You must declare the object before you can use it.

ACTION: Add or correct the declaration for the object, and make sure the declaration is in the proper place in the VHDL Design File.

VHDL error at <*location*>: object cannot be indexed because it has <*type*> type rather than array type

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you specified an index of a object. However, in a Signal Declaration or Variable Declaration, you declared that the object has the specified type and not an array type. You can specify an index of a object only if the object has an array type.

ACTION: Remove the index for the object, or declare the object so it has an array type.

See also:

Sections 3.2.1, 4.3.1.2, and 4.3.1.3 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: operator <*name*> cannot be used for value

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the specified operator for a value. However, you cannot use the operator for this value.

ACTION: Change the operator to one that you can use for the value.

See also:

Section 7.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: range direction of object slice must be same as range direction of object

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you specified a slice of an object that has a subtype of a one-dimensional array type. However, the direction of the range you specified for the slice is different than the direction of the object range, which you specified in a Subtype Declaration. For example, the slice may have an ascending range and the object may have a descending range. The range of the slice and the range of the object must both be ascending or descending.

ACTION: Change the direction of the slice range so it is the same as the direction of the object range.

See also:

Sections 3, 4.2, and 6.5 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: range direction of object slice must be same as range direction of object

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you specified a slice of an object that has a one-dimensional array type. However, the direction of the range you specified for the slice is different than the direction of the object range, which you specified in a Type Declaration. For example, the slice may have an ascending range and the object may have a descending range. The range of the slice and the range of the object must both be ascending or descending.

ACTION: Change the direction of the slice range so it is the same as the direction of the object range.

See also:

Sections 3, 4.1, and 6.5 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: real type object <*name*> cannot be used in operations

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the specified object as an operand in an operation. However, you declared a real type for the object in a Constant Declaration, Signal Declaration, or Variable Declaration. You cannot use real type objects as operands in operations

ACTION: Declare a different type for the object, or remove the real type object from the operation.

See also:

Sections 3.1.4.1 and 4.3.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: real type objects cannot be used in operations

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used one or more objects as operands in an operation. However, you declared a real type for the object(s) in one or more Constant Declarations, Signal Declarations, or Variable Declarations. You cannot use real type objects as operands in operations.

ACTION: Declare a different type for the object(s), or remove the real type object(s) from the operation.

See also:

Sections 3.1.4.1 and 4.3.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: record element <*name*> must have <*type*> type

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you accessed the specified element of a record type object. However the type you specified for the record element in the record type's Type Declaration is not the same as the type that is required at the location where you accessed the element. For example, you may have accessed a record element with an array type in an expression, but the expression requires a record element with an integer type.

ACTION: Change the type of the record element so it is the same as the type required at the location the element is accessed, or access a different record element with the required type.

See also:

Section 3.2.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: record type <*type*> is used but not declared

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the specified record type. However, you did not declare the record type in an Type Declaration. You must declare a record type before you can use it.

ACTION: Replace the type with a previously defined record type, or define the record type in a Type Declaration. However, do not declare an record type with a name that you already used for a different construct.

See also:

Section 3.2.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: record type object <*name*> must contain element <*name*>

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you accessed the specified element of the specified record type object. However, you did not declare the element in the Type Declaration for the object's record type. To access an element of a record type object, you must first declare the element in the Type Declaration for the record type.

ACTION: Remove the reference to the element, or add the element to the Type Declaration for the object's record type.

See also:

Sections 3.2.2 and 4.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: record type object <*name*> must contain element <*name*>

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you accessed the specified element of the specified record type object. However, you did not declare the element in the Type Declaration for the object's type. To access an element of an object with a record type, you must first declare the element in the Type Declaration for the record type.

ACTION: Remove the reference to the element, or add the element to the Type Declaration for the object's record type.

See also:

Sections 3.2.2 and 4.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: resolution function <*name*> is used but not declared

---

CAUSE: In Subtype Declaration or Signal Declaration at the specified location in a VHDL Design File (**.vhd**), you used the specified resolution function. However, you did not declare the resolution function in a Subprogram Declaration. You must declare a resolution function before you can use it.

ACTION: Replace the resolution function with a previously defined resolution function, or define the resolution function in a Subprogram Declaration. However, do not define a resolution function with the name you used in the Subtype Declaration or Signal Declaration if you already used the name for a different construct.

See also:

Sections 2.1, 2.4, 4.2, and 4.3.1.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: right bound of slice must belong to range of corresponding object

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you specified a slice of an object that has an array type. However, the right bound of the range you specified for the slice exceeds the range of the array type object. The right bound of the slice must belong to the object range.

ACTION: Change the right bound of the slice so the bound belongs to the range of the object.

See also:

Section 6.5 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: second actual parameter of function <*name*> must be constant

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you specified actual parameters for a function. However, the second parameter you specified is not constant. The second parameter must be constant because you specified `FEED_THROUGH` for the `SYNTHESIS_RETURN` attribute in the function's Subprogram Body.

ACTION: Change the second parameter so it is constant.

See also:

Section 2.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: selected name <*name*> must have <*type*> type

---

CAUSE: At the specified location in a VHDL Design File (**.vhd**), you accessed the specified selected name. However, the selected name type is not the same as the specified type, which is required at the location where you accessed the selected name. The selected name type must match the type required at the location.

ACTION: Change the type of the selected name so that it is the same as the type required at the location where you accessed the selected name.

See also:

Section 6.3 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: signal <*name*> is not synthesizable because it does not hold its value after a clock edge

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the specified signal. However, you assigned a value to the signal during a clock edge, and then assigned a different value to the signal immediately after the clock edge. Because the signal cannot hold the value assigned to it during the clock edge, the signal is not synthesizable. For example, the If Statement in the following code assigns one value to the signal `state` during the clock edge, and then assigns another value to `state` immediately after the clock edge:

```
PROCESS(clk)
BEGIN
   IF clk='1' AND clk'event THEN
      state <= write;
   ELSE
      state <= idle;
   END IF;
END PROCESS
```

ACTION: Make sure you do not assign a value to a signal on both the clock edge and another condition of the signal. For the example above, you can use the following code:

```
PROCESS(rst,clk)
BEGIN
   IF rst='1' THEN
      state <= idle;
   ELSIF clk='0' AND clk'event THEN
      state <= write;
   END IF;
END PROCESS
```

See also:

Sections 8.4 and 8.7 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: signature profile used for attribute, entity, or alias near <*text*> must match expected parameter or result type <*type*>

CAUSE: In an attribute name, entity designator, or alias declaration at the specified location in a VHDL Design File (**.vhd**) and near the specified text, you used a signature to distinguish between the overloaded functions, overloaded procedures, or overloaded enumeration literals that an attribute, entity, or alias uses. However, the signature profile does not match the specified expected parameter or result type for one or more of the functions, procedures, or enumeration literals. The signature profile must match all the expected parameter or result types.

ACTION: Change the signature profile so it matches all the expected parameter or return types for the overloaded functions, procedures, or enumeration literals.

See also:

Section 2.3.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: slice of object cannot be specified for object that has an array type of more than one dimension

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you specified a slice of an object that has an array type with two or more dimensions. For example, the following code contains a slice of the signal `mem_inst`, which has a two-dimensional array type:

```
ARCHITECTURE ex1 OF example IS
    TYPE mem_cell IS ARRAY (0 to 1023, 0 to 31) of BIT;
    SIGNAL mem_inst : mem_cell;
BEGIN
    o(0 to 7) <= mem_inst(0 to 1, 0 to 3);
END ex1;
```

However, you can specify a slice only for an object that has a one-dimensional array type.

ACTION: Replace the slice with an enumeration of the array values you want to specify. For example:

```
ARCHITECTURE ex1 OF example IS
    TYPE mem_cell IS ARRAY (0 to 1023, 0 to 31) of BIT;
    SIGNAL mem_inst : mem_cell;
BEGIN
    o(0) <= mem_inst(0,0);
    o(1) <= mem_inst(0,1);
    o(2) <= mem_inst(0,2);
    o(3) <= mem_inst(0,3);
    o(4) <= mem_inst(1,0);
    o(5) <= mem_inst(1,1);
    o(6) <= mem_inst(1,2);
    o(7) <= mem_inst(1,3);
END ex1;
```

See also:

Sections 3 and 6.5 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: slice that is assigned to target slice has <*number*> elements, but must have same number of elements as target slice (<*number*>)

CAUSE: In an assignment at the specified location in a VHDL Design File (**.vhd**) you assigned a slice of an array type object to the slice of another array type object. However, as specified in the message text, the number of elements in the two slices are not the same. For example, in the following Signal Assignment Statement, the slice of signal `s` contains 8 elements, and the slice of signal `i` contains 16 elements:

```
s(0 to 7) <= i(0 to 15);
```

The slice of the object you are assigning and the slice of the object to which you are making the assignment must have the same number of elements.

ACTION: Change one or both slices so that the slices contain the same number of elements. For the example above, you can change the slice of `s` to `s(0 to 15)`, or change the slice of `i` to `i(0 to 7)`.

See also:

Sections 6.5, 8.4, and 8.5 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: subprogram or value <*name*> is used for, but not declared in, Package Body <*name*>

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the specified subprogram or value as part of the specified Package Body. However, you did not declare the subprogram or value in the Package Body. You must declare a subprogram or value in a Package Body before you can use it for the Package Body.

ACTION: Make sure you specify the correct Package Body for the subprogram or value. Or, replace the subprogram or value with one that you declared in the Package Body.

See also:

Sections 2.6 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: subtype indication cannot contain <>

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the undefined range symbol (<>) in a subtype indication. However, the subtype indication must have a defined range for the subtype. For example, you may have already constrained the subtype.

ACTION: Change the undefined range symbol in the subtype indication to a defined range.

See also:

Sections 3.2.1 and 4.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: type of identifier <*name*> does not agree with its usage as <*type*> type

CAUSE: In a VHDL Design File (.vhd) at the specified location, you used the specified identifier. However, the identifier does not have the specified type, which is required for the identifier's usage. For example, the Signal Assignment Statement in the following code assigns the value x, which has an integer type, to the signal o, which has a bit type:

```
ENTITY err IS
   PORT
   (
      clk   : IN   BIT;
      i     : IN   BIT;
      o     : OUT  BIT
   );
END err;


ARCHITECTURE a OF err IS
   CONSTANT x: INTEGER := 5;
BEGIN
   PROCESS(clk)
   BEGIN
      o <= x;
   END PROCESS;
END a;
```

The value x must have a bit type to match the type of the signal to which the value is assigned.

This error can also occur when the actual parameter you use in a Function Call or Procedure Call Statement does not have the same type as the corresponding formal parameter in the function or procedure. The text of the message specifies the type that the actual parameter must have. In addition, this error can occur when you use an identifier as a Boolean value, but you use a non-Boolean value for the identifier.

ACTION: Change either the type or usage of the identifier so the type you use for the identifier matches the expected type. In the example above, you can change the type of value x to bit (and change the initial value of the constant accordingly), change the type of signal o to integer, or remove or change the Signal Assignment Statement so it does not assign x to o.

See also:

Section 3 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: type of identifier <*name*> does not agree with its usage as <*type*> type

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the specified identifier. However, the identifier does not have the specified type, which is required for the identifier's usage. For example, the Signal Assignment Statement in the following code assigns the value x, which has an integer type, to the signal o, which has a bit type:

```
ENTITY err IS
   PORT
   (
      clk   : IN   BIT;
      i     : IN   BIT;
      o     : OUT  BIT
   );
END err;


ARCHITECTURE a OF err IS
   CONSTANT x: INTEGER := 5;
BEGIN
   PROCESS(clk)
   BEGIN
      o <= x;
   END PROCESS;
END a;
```

The value x must have a bit type to match the type of the signal to which the value is assigned.

This error can also occur when you use an identifier as a Boolean value, but you use a non-Boolean value for the identifier.

ACTION: Change the type or usage of the identifier. In the example above, you can change the type of value x to bit, change the type of signal o to integer, or remove or change the Signal Assignment Statement so it does not assign x to o.

See also:

Section 3 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: unit name is illegal

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used a unit name. However, the unit name is illegal.

ACTION: Make sure the name is legal. Refer to 5.2.1.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual* for more information on the restrictions for unit names. If necessary, correct the context or scope in which the unit name is being used.

VHDL error at <*location*>: value cannot be assigned to constant <*name*>

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you assigned a value to the specified constant. However, you cannot assign a value to a constant.

ACTION: Remove the assignment to the constant. If you want, assign the value to a valid object, such as a variable.

See also:

Sections 4.3 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: value cannot be assigned to item <*text*> because item is not an object

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you assigned a value to the specified item. However, the item is not an object; for example, it is a type, entity, or component. You cannot assign a value to a non-object item.

ACTION: Remove the assignment to the item. If you want, assign the value to a valid object. If the item is a type, you can declare an object that has the type, and then assign a value to the object.

See also:

Section 4.3 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: value covered by choice <*name*> is already covered by another choice

---

CAUSE: In a construct at the specified location in a VHDL Design File (**.vhd**), you used the specified choice for a value. However, another choice in the construct already covers the value.

ACTION: Remove one of the choices for the value.

See also:

Sections 7.3.2, 8.8, and 9.5.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: value is assigned to index of object <*name*>, but object index must belong to object index range

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you assigned a value to an index of the specified array type object. However, the target index you specified for the object is not within the bounds of the index range that you specified for the object. For example, the Signal Assignment Statement in the following code assigns a value to the indexed object `o(4)`, but the index `4` does not belong to the index range of target signal `o`, which is `(0 to 3)`:

```
ENTITY example IS

   PORT
   (
      clk       : IN     std_logic;
      i         : IN     std_logic_vector (0 to 3);
      o         : OUT    std_logic_vector (0 to 3)
   );

END example;

ARCHITECTURE ex1 OF example IS
BEGIN
   o(4) <= i(3);
END ex1;
```

The target index of the object must belong to the index range you specify for the object.

ACTION: Change the target index for the object, or change the bounds of the object index range to include the index. In the example above, you must change `o(4)` to `o(0)`, `o(1)`, `o(2)`, or `o(3)`; or change the index range `(0 to 3)` to include the target index `4`.

See also:

Sections 3.2.1, 4.3.1, 6.4, 8.4, and 8.5 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: values cannot be assigned to only a section of a port with an undefined range

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you specified a port with an undefined range using the undefined range symbol (<>). You then attempted to assign values to only a section of the port's undefined range. However, when a port has an undefined range, you must assign values only to the entire port.

ACTION: Specify a range for the port, or assign values to the entire port.

See also:

Sections 1.1.1.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>: variable must be constrained

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used an unconstrained variable. However, in the context in which the variable is being used, the variable must be constrained.

ACTION: Changed the unconstrained variable to a constrained variable.

See also:

Sections 4.3.1.3, 4.3.2.2, and 8.5 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error at <*location*>:Can't access enumeration literal to the right of enumeration literal <*name*> in enumeration type <*name*>

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you attempted to access an enumeration literal to the right of the specified enumeration literal in the specified enumeration type. However, the specified enumeration literal is the last enumeration literal in the enumeration type, and therefore it has no enumeration literals to its right for Integrated Synthesis to access. For example, the `'SUCC` predefined attribute in the following code attempts to access an enumeration literal that is positioned to the right of enumeration literal `st2` in the enumeration type `StateType`. However, `st2` is the last enumeration literal in `StateType`.

```
TYPE StateType IS (st0, st1, st2);
SIGNAL current_state: StateType;
SIGNAL next_state: StateType;

PROCESS(i, current_state)
BEGIN
   current_state <= st2;
   CASE current_state is
      WHEN st2 =>
         next_state <= StateType'SUCC(st2);
      WHEN others =>
         next_state <= st2;
   END CASE;
END PROCESS;
```

ACTION: Make sure you do not try to access an enumeration literal that is positioned to the right of the last enumeration literal in an enumeration type.

See also:

Section 3.1.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL error: value <*text*> cannot contain divisor of zero

CAUSE: The specified value in a VHDL Design File (**.vhd**) contains a divisor that is zero or an expression that reduces to zero. However, a value cannot be divided by zero.

ACTION: Make sure the divisor is not zero or an expression that reduces to zero.

VHDL Exit Statement error at <*location*>: Exit Statement must be inside Loop Statement for loop <*name*>

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used an Exit Statement. However, you did not use the Exit Statement in the Loop Statement for the specified loop. The Exit Statement must be inside the Loop Statement.

ACTION: Move the Exit Statement inside the Loop Statement, or expand the Loop Statement so it includes the Exit Statement.

See also:

Sections 8.9 and 8.11 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Exit Statement or Next Statement error at <*location*>: loop name <*name*> is used in Exit Statement or Next Statement, but not specified in Loop Statement

---

CAUSE: In an Exit Statement or a Next Statement at the specified location in a VHDL Design File (**.vhd**), you used the specified loop name. However, the loop name is not the same as the loop name you specified in a Loop Statement that contains the Exit Statement or Next Statement. The loop name in an Exit Statement or a Next Statement must be the same as the loop name of one of the Loop Statements that contains the Exit Statement or Next Statement.

ACTION: Change the loop name of the Exit Statement or Next Statement to match the loop name of one of the Loop Statements.

See also:

Sections 8.9, 8.10, and 8.11 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Exit Statement, Next Statement, or Return Statement error at <*location*>: edge-triggered or event-triggered condition cannot be used in Exit Statement, Next Statement, or Return Statement

---

CAUSE:	In a Exit Statement, Next Statement, or Return Statement at the specified location in a VHDL Design File (**.vhd**), you used an an edge-triggered or event-triggered condition. However, you cannot use an edge-triggered or event-triggered condition in these constructs.
ACTION:	Remove the edge-triggered or event-triggered condition from the Exit Statement, Next Statement, or Return Statement.

See also:

Sections 8.10, 8.11, and 8.12 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL expression error at <*location*>: <*text*> operator cannot be used for non-constant values

---

CAUSE: In an expression at the specified location in a VHDL Design File (**.vhd**), you used the specified operator for values that are not constant. However, you cannot use the operator for operands that are non-constant values.

ACTION: Change the operator to one that can be used with non-constant values.

See also:

Section 7.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL expression error at <*location*>: <*type*> operator cannot be used for integer values

---

CAUSE: In an expression at the specified location in a VHDL Design File (**.vhd**), you used the specified operator for integer values. However, you cannot use the operator for operands that are integer values.

ACTION: Change the operator to one that you can use with integer values.

See also:

Sections 3.1.2 and 7.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL expression error at <*location*>: <*type*> operator cannot be used for real values

---

CAUSE: In an expression at the specified location in a VHDL Design File (**.vhd**), you used the specified operator for real values. However, you cannot use an operator for operands that are real values.

ACTION: Change the operator to one that you can use with real values.

See also:

Sections 3.1.4 and 7.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL expression error at <*location*>: expression has <*number*> elements, but must have <*number*> elements

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used an expression with the specified number of elements. However, the expression must have a different number of elements, which is specified in the message text.

ACTION: Make sure the expression uses the required number of elements.

See also:

Section 7 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL expression error at <*location*>: illegal <*text*> in expression

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the specified item in an expression. However, the item is illegal for this expression.

ACTION: Remove the item, or make sure the item is legal.

See also:

Section 7 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL expression error at <*location*>: operands for operator <*text*> must have equal length

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you specified an expression. However, the operands for the specified operator in the expression have different lengths. The operands must have equal lengths.

ACTION: Make sure all the operands for the operator have the same length.

See also:

Section 7 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Function Call or Component Instantiation error at <*location*>: recursive function or recursive design entity <*name*> is not supported

---

CAUSE:	In a Function Call or Component Instantiation at the specified location in a VHDL Design File (**.vhd**), you used the specified recursive function or design entity. However, Integrated Synthesis does not support recursive functions or recursive design entities.
ACTION:	Remove the Function Call or Component Instantiation. Or, change the Function Call or Component Instantiation so it refers to a function or design entity that does not involve recursion.

See also:

Sections 7.3.3 and 9.6 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Function Call or Procedure Call Statement error at <*location*>: calls to function or procedure <*name*> caused stack overflow

---

CAUSE:	In a Function Call or Procedure Call Statement at the specified location in a VHDL Design File (**.vhd**), you called the specified function or procedure. However, the function or procedure call pushed too many items on the stack and therefore caused a stack overflow.
ACTION:	Change the Function Call or Procedure Call Function so it does not cause a stack overflow.

See also:

Sections 2.1, 7.3.3, and 8.6 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Function Call or Procedure Call Statement error at <*location*>: Function Call or Procedure Call statement for function or procedure <*name*> must contain <*number*> actual parameters

---

CAUSE:	In a Function Call or Procedure Call Statement at the specified location in a VHDL Design File (**.vhd**), you specified actual parameters for the formal parameters of the specified function or parameter. However, the Function Call or Procedure Call Statement must contain one actual parameter for each formal parameter that you specified for the function or parameter in a Subprogram Declaration. The text of the message specifies the number of actual parameters the Function Call or Procedure Call Statement must contain.
ACTION:	Make sure the Function Call or Procedure Call Statement contains the correct number of actual parameters.

See also:

Sections 2.2, 7.3.3, and 8.6 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Function Call or Procedure Call Statement error at <*location*>: object class of actual parameter <*name*> must match object class <*type*> of formal parameter <*name*>

CAUSE: In a Function Call or Procedure Call Statement at the specified location in a VHDL Design File (**.vhd**), you specified an actual parameter for the specified formal parameter of a function or procedure. However, the actual parameter does not have the same specified class (that is, `signal`, `variable`, `constant`, or `file`) as the formal parameter. The object class of the actual parameter and the object class of the formal parameter must match.

ACTION: Change the object class of the actual parameter or the formal parameter so the object classes match.

See also:

Sections 2.1, 2.1.1, and 2.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Generate Statement error at <*location*>: condition must be constant

---

CAUSE: In a Generate Statement at the specified location in a VHDL Design File (**.vhd**), you used a condition that is not constant. However, the condition must be constant.

ACTION: Change the condition so it is constant.

See also:

Section 9.7 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Generic Map Aspect error at <*location*>: formal generic <*name*> in named Association List must have corresponding generic in block <*name*>

CAUSE: In a Generic Map Aspect at the specified location in a VHDL Design File (**.vhd**), you used a named Association List to associate actual generics with the formal generics in the specified block. However, the block does not have a formal generic with the same name as the specified formal generic in the Generic Map Aspect. A formal generic in a Generic Map Aspect must have a corresponding formal generic in the associated block.

ACTION: Make sure the Generic Map Aspect only associates actual generics to formal generics that exist in the associated block.

See also:

Section 4.3.2.2, 5.2.1.2, 9.1, and 12.2.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Generic Map Aspect error at <*location*>: ignored illegal actual generic literal value <*name*> that is associated with formal generic <*name*>

CAUSE: In a Generic Map Aspect at the specified location in a VHDL Design File (**.vhd**), you used a named or positional Association List to associate the specified actual generic literal value to the specified formal generic of a block. However, the literal value of the actual generic is illegal, and therefore Integrated Synthesis ignored the value when elaborating the Generic Map Aspect.

ACTION: Correct the actual generic so it is a legal literal value.

See also:

Section 12.2.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Generic Map Aspect error at <*location*>: positional Association List contains <*number*> actual generics for block <*name*>, but block has different number of formal generics

---

CAUSE: In a Generic Map Aspect at the specified location in a VHDL Design File (**.vhd**), you used a positional Association List to associate actual generics with the formal generics of the specified block. However, the positional Association List contains the specified number of actual generics, which is not the same as the number of formal generics in the block. The number of actual generics in the positional Association List must be the same as the number of formal generics in the block. The submessage of this message indicates the location of the construct that specifies the formal generics for the block.

ACTION: Add or remove actual generics in the positional Association List so that the number of actual generics matches the number of formal generics in the block. To avoid receiving this message in the future, replace the positional Association List with a named Association List.

See also:

Sections 4.3.2.2, 5.2.1.2, 9.1, and 12.2.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Generic Map Aspect error at <*location*>: positional Association List contains <*number*> actual generics for block <*name*>, but block has different number of formal generics

---

CAUSE: In a Generic Map Aspect at the specified location in a VHDL Design File (**.vhd**), you used a positional Association List to associate actual generics with the formal generics of the specified block. However, the positional Association List contains the specified number of actual generics, which does not match the number of formal generics in the block. The number of actual generics in the positional Association List must be the same as the number of formal generics in the block.

ACTION: Add or remove actual generics in the positional Association List so that the number of actual generics matches the number of formal generics in the block. Make sure that the actual generics are ordered so each corresponds to the correct formal generic. To avoid receiving this message in the future, replace the positional Association List with a named Association List.

See also:

Sections 4.3.2.2, 5.2.1.2, 9.1, and 12.2.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Generic Map Aspect error at <*location*>: positional Association List contains too many actual generics

---

CAUSE: In a Generic Map Aspect at the specified location in a VHDL Design File (**.vhd**), you used a positional Association List to associate actual generics with the formal generics of a block or component. However, the positional Association List contains more actual generics than there are formal generics in the block or component. The number of actual generics in the positional Association List must be the same as the number of formal generics in the block or component.

ACTION: Remove the extra actual generics from the positional Association List. Make sure that the actual generics are ordered so each corresponds to the correct formal generic. To avoid receiving this message in the future, replace the positional Association List with a named Association List.

See also:

Section 4.3.2.2 and 5.2.1.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL information at <*location*>: object <*name*> is never assigned

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you specified an object that is either a variable in a process or an item in an array. However, the design never assigns a value to the object; that is, the design never uses the object on the left-hand side (LHS) of an expression. If the object is not used at a later stage in the design process (for example, when stitching different parts of the design together), the object is synthesized as a constant net.

ACTION: If you want the object to be synthesized as a constant net, no action is required. Otherwise, make sure the object is used at a later stage in the design process, or remove the object.

See also:

The IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL information at <*location*>: object <*name*> is never used

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you specified an object that is either a variable in a process or an item in an array. However, the design never uses the object on the right-hand side (RHS) of an expression. If the object is not used at a later stage in the design process (for example, when stitching different parts of the design together), and if the object is not eliminated during synthesis, the object becomes a floating net.

ACTION: Make sure the object is used at a later stage in the design process, or is eliminated during synthesis. If necessary to prevent the object from becoming a floating net, remove the object.

See also:

The IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL information: block <*name*> is declared at <*location*>

CAUSE: This message indicates the location of the construct that declares the specified block. The construct specifies the generics and ports for the block. Examples of constructs are Block Statements, Component Declarations, Entity Declarations, or instantiations of blocks, components, or entities. This message is part of another message that appears above this message in the Messages window and in the Messages section of the Report window.

ACTION: No action is required.

VHDL information: ignored user-defined design library *<name>*

CAUSE: In the MAX+PLUS II Advanced Synthesis software, you attempted to compile a project that contains a user-defined design library that you created with the MAX+PLUS II Advanced Synthesis software version 2.0 or earlier. However, in the MAX+PLUS II Advanced Synthesis software version 2.1 and later, you cannot include individual VHDL Design Files (**.vhd**) in a project using user-defined design libraries. Instead, you must analyze, compile, and otherwise process all the VHDL Design Files in a project at one time. As a result, the MAX+PLUS II Advanced Synthesis software ignored the user-defined design library you attempted to include in the project.

ACTION: Add all the project's VHDL Design Files to the project. If you have already added all the project's VHDL Design Files to the project, no action is required.

See also:

Adding & Removing Files in a Project

VHDL Interface Declaration error at <*location*>: constant <*name*> must be of mode IN in Interface Declaration

---

CAUSE: In an Interface Declaration at the specified location in a VHDL Design File (**.vhd**), you declared the specified constant. However, you specified a mode for the constant that is not mode `IN`. A constant in an Interface Declaration must be of mode `IN`.

ACTION: Change the mode of the constant to `IN`.

See also:

Section 4.3.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Interface Declaration error at <*location*>: constant or signal <*name*> in Interface Constant or Signal Declaration cannot be subtype <*type*> of file type or access type

---

CAUSE: In an Interface Constant Declaration or Interface Signal Declaration at the specified location in a VHDL Design File (**.vhd**), you declared the specified interface constant or signal with the specified subtype of a file type or access type. However, the constant or signal cannot have a subtype of a file type or access type.

ACTION: Change the subtype indication for the interface constant or signal so it does not denote a file or access subtype, or change the Interface Constant Declaration or Interface Signal Declaration to an Interface File Declaration.

See also:

Section 4.3.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Interface Declaration error at <*location*>: formal <*signal*> parameter <*name*> in Interface Signal Declaration cannot have default expression

---

CAUSE: In an Interface Signal Declaration at the specified location in a VHDL Design File (**.vhd**), you used a default expression for the specified formal signal parameter. However, an Interface Declaration for a formal signal parameter cannot contain a default expression.

ACTION: Remove the default expression.

See also:

Section 4.3.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Interface Declaration error at <*location*>: formal variable parameter <*name*> of mode <*text*> in Interface Variable Declaration cannot have default expression

---

CAUSE: In an Interface Variable Declaration at the specified location in a VHDL Design File (**.vhd**), you used a default expression for the specified formal variable parameter of the specified mode. However, only a formal variable parameter of mode `IN` can have a default expression.

ACTION: Remove the default expression, or change the mode of the formal variable parameter to `IN`.

See also:

Section 4.3.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Interface Declaration error at <*location*>: Interface File Declaration cannot contain mode or default expression

---

CAUSE: In an Interface File Declaration at the specified location in a VHDL Design File (**.vhd**), you used a mode and/or default expression. However, an Interface File Declaration cannot contain a mode or default expression.

ACTION: Remove the mode and/or default expression from the Interface File Declaration.

See also:

Section 4.3.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Interface Declaration error at <*location*>: interface object <*text*> of mode LINKAGE cannot have default expression

---

CAUSE: In an Interface Declaration at the specified location in a VHDL Design File (**.vhd**), you used a default expression for the specified interface object of mode `LINKAGE`. However, an interface object of mode `LINKAGE` cannot have a default expression.

ACTION: Change the mode of the interface object to `IN`, `OUT`, `INOUT`, or `BUFFER`, whichever is appropriate; or remove the default expression.

See also:

Section 4.3.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Interface Declaration error at <*location*>: subtype indication of formal <*name*> in Interface File Declaration must denote file subtype

---

CAUSE:	In an Interface File Declaration at the specified location in a VHDL Design File (**.vhd**), you used a subtype indication for the specified formal that is not of a file type. However, the subtype indication must denote a subtype of a file type.
ACTION:	Change the subtype indication so it denotes a subtype of a file type.

See also:

Section 4.3.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Interface Declaration error in <*location*>: interface object <*name*> of mode out cannot be read. Change object mode to buffer or inout.

CAUSE: In an Interface Declaration at the specified location in a VHDL Design File (**.vhd**), you declared the specified interface object with a mode of `OUT`. Integrated Synthesis attempted to read the value of the interface object, but cannot do so because the interface object has the mode `OUT`.

ACTION: Change the mode of the interface object to `inout` or `buffer`, or use a temporary internal object for the interface object. The following example shows how you can use a temporary internal object:

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY example IS
   PORT
   (
      ena   : IN   BIT;
      i     : IN   BIT;
      o     : OUT  BIT
   );
END example;

ARCHITECTURE a OF example IS
   SIGNAL tmp : BIT;
BEGIN
   PROCESS (ena, i)
   BEGIN
   CASE ena IS
      WHEN '1' =>
         tmp  <= i;
      WHEN OTHERS =>
         tmp  <= tmp;
   END CASE;
   END PROCESS;
   o <= tmp;
END a;
```

See also:

Section 4.3.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Interface List error at <*location*>: identifier <*name*> must be a <*constant or signal*>

CAUSE 1:	In a generic Interface List at the specified location in a VHDL Design File (**.vhd**), you used the specified identifier, which is not a constant. A generic Interface List must consist entirely of interface constant declarations.
ACTION:	Change the identifier so it is a constant.
CAUSE 2:	In a port Interface List at the specified location in a VHDL Design File (**.vhd**), you used the specified identifier, which is not a signal. A port Interface List must consist entirely of interface signal declarations.
ACTION:	Change the idenfier so it is a signal.

See also:

Section 4.3.2.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Loop Statement error at <*location*>: infinite loops are not supported for synthesis.

CAUSE: In a Loop Statement at the specified location in a VHDL Design File (**.vhd**), you specified a loop that does not terminate. Integrated Synthesis does not support infinite loops for synthesis. This message may occur because of an error in the Loop Statement's `WHILE` iteration scheme, `FOR` iteration scheme, or Exit Statement; or because you did not increment the variable for the loop.

ACTION: Make sure that the Loop Statement specifies a loop that terminates.

See also:

Sections 8.9 and 8.11 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Loop Statement error at <*location*>: loop must terminate at or before 1000 iterations

---

CAUSE: In a Loop Statement at the specified location in a VHDL Design File (**.vhd**), you specified a loop that does not terminate after 1000 iterations. This message may occur because of an error in the Loop Statement's `WHILE` iteration scheme, `FOR` iteration scheme, or Exit Statement; or because you did not increment the variable for the loop.

ACTION: Make sure that the Loop Statement specifies a loop that terminates in 1000 or less iterations.

See also:

Section 8.9 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Loop Statement error at <*location*>: WHILE iteration scheme condition cannot contain signals

---

CAUSE: In a Loop Statement at the specified location in a VHDL Design File (**.vhd**), you used a `WHILE` iteration scheme condition that contains one or more signals. However, the condition must be an expression of integer variables that contains no signals.

ACTION: Remove any signals from the `WHILE` iteration scheme condition.

See also:

Section 8.9 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Next Statement error at <*location*>: Next Statement must be inside Loop Statement

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used a Next Statement. However, the Next Statement is not inside a Loop Statement, and therefore does not apply to a specific Loop Statement. The Next Statement must be inside the Loop Statement to which the Next Statement applies.

ACTION: Place the Next Statement inside the Loop Statement to which the Next Statement applies or, if there is no corresponding Loop Statement, remove the Next Statement.

See also:

Sections 8.9 and 8.10 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Next Statement error at <*location*>: Next Statement must be inside Loop Statement for loop <*name*>

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used a Next Statement. However, you did not use the Next Statement in the Loop Statement for the specified loop. The Next Statement must be inside the Loop Statement.

ACTION: Move the Next Statement inside the Loop Statement, or expand the Loop Statement so it includes the Next Statement.

See also:

Sections 8.9 and 8.10 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Package Body error at <*location*>: package <*name*> is used but not declared

---

CAUSE: In a Package Body at the specified location in a VHDL Design File (**.vhd**), you used the specified package. However, you did not declare the package in a Package Declaration. You must declare a package before you can use it.

ACTION: Replace the package with a previously defined package, or define the package in a Package Declaration. However, do not define a package with the name you used in the Package Body if you already used the name for a different construct.

See also:

Sections 2.5 and 2.6 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Package Body error at <*location*>: package <*name*> is used but not declared

---

CAUSE: In a Package Body at the specified location in a VHDL Design File (**.vhd**), you used the specified package. However, you did not declare the package in a Package Declaration. You must declare a package before you can use it.

ACTION: Replace the package with a previously defined package, or define the package in a Package Declaration. However, do not define a package with the name you used in the Package Body if you already used the name for a different construct.

See also:

Sections 2.5 and 2.6 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Port Map Aspect error at <*location*>: positional Association List contains <*number*> actual ports for block <*name*>, but block has different number of formal ports

---

CAUSE: In a Port Map Aspect at the specified location in a VHDL Design File (**.vhd**), you used a positional Association List to associate actual ports with the formal ports of the specified block. However, the positional Association List contains the specified number of actual ports, which is not the same as the number of formal ports in the block. The number of actual ports in the positional Association List must be the same as the number of formal ports in the block. The submessage of this message indicates the location of the construct that specifies the formal ports for the block.

ACTION: Add or remove actual ports in the positional Association List so that the number of actual ports matches the number of formal ports in the block. To avoid receiving this message in the future, replace the positional Association List with a named Association List.

See also:

Sections 5.2.1.2 and 9.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Port Map Aspect error at <*location*>: positional Association List contains too many actual ports

---

CAUSE: In a Port Map Aspect at the specified location in a VHDL Design File (**.vhd**), you used a positional Association List to associate actual ports with the formal ports of a block or component. However, the positional Association List contains more actual ports than there are formal ports in the block or component. The number of actual ports in the positional Association List must be the same as the number of formal ports in the block or component.

ACTION: Remove the extra actual ports from the positional Association List. Make sure that the actual ports are ordered so each corresponds to the correct formal port. To avoid receiving this message in the future, replace the positional Association List with a named Association List.

See also:

Section 4.3.2.2 and 5.2.1.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Process Statement error at <*location*>: Process Statement must contain either a sensitivity list or a Wait Statement

---

CAUSE:	In a Process Statement at the specified location in a VHDL Design File (**.vhd**), you did not use either a sensitivity list or a Wait Statement. However, the Process Statement must contain a sensitivity list or a Wait Statement.
ACTION:	Add either a sensitivity list or a Wait Statement to the Process Statement.

See also:

Sections 8.1 and 9.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Process Statement error at <*location*>: can't synthesize condition that contains event expression(s) and more than one signal

---

CAUSE: In a Process Statement at the specified location in a VHDL Design File (**.vhd**), you used a condition that contains one or more event expressions and more than one signal. Integrated Synthesis cannot synthesize conditions with event expression(s) and more than one signal. For example, the If Statement in the following code contains conditions that Integrated Synthesis cannot synthesize because they each contain event expressions with the signals `ffclk` and `ffin`:

```
myrff : PROCESS (ffin, ffrst, ffclk)
BEGIN
   IF ffrst = '1' THEN
      ffout <= '0';
   ELSIF rising_edge(ffclk) AND ffin = '1' THEN
      ffout <= '1';
   ELSIF rising_edge(ffclk) AND ffin = '0' THEN
      ffout <= '0';
   END IF;
END PROCESS;
```

ACTION: Replace the condition with a nested condition. For the example above, you can rewrite the If Statement so its conditions each have an event expression and only one signal:

```
myrff : PROCESS (ffin, ffrst, ffclk)
BEGIN
   IF ffrst = '1' THEN
      ffout <= '0';
   ELSIF rising_edge(ffclk) THEN
      IF ffin = '1' THEN
        ffout <= '1';
      ELSE
        ffout <= '0';
      END IF;
   END IF;
END PROCESS;
```

See also:

Sections 8.7 and 9.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Process Statement error at <*location*>: Process Statement cannot contain both a sensitivity list and a Wait Statement

---

CAUSE:	In a Process Statement at the specified location in a VHDL Design File (**.vhd**), you used both a sensitivity list and a Wait Statement. However, because the sensitivity list acts as an implicit Wait Statement, the Process Statement cannot contain both a sensitivity list and a Wait Statement.
ACTION:	Remove either the sensitivity list or the Wait Statement from the Process Statement.

See also:

Sections 8.1 and 9.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Process Statement error at <*location*>: Process Statement must contain only one Wait Statement

---

CAUSE: In a Process Statement at the specified location in a VHDL Design File (**.vhd**), you used more than one Wait Statement. However, a Process Statement must contain only one Wait Statement.

ACTION: Remove all but one Wait Statement from the Process Statement. If you want, create new Process Statements for each Wait Statement you removed.

See also:

Section 8.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Process Statement warning at <*location*>: signal <*name*> is in sequential statement, but is not in sensitivity list

---

CAUSE: In a Process Statement at the specified location in a VHDL Design File (**.vhd**), you used the specified signal as an input signal. However, you did not specify the signal in the Process Statement's sensitivity list. For example, the Process Statement in the following code uses the signal `i` as in input signal, but the Process Statement's sensitivity list does not contain the signal:

```
PROCESS(clk)
BEGIN
   o <= i;
END PROCESS;
```

Because it causes the process to not execute when the input signal changes, the omission may cause an error during the future processing of the design. In the example above, an error may occur because the process executes only when `clk` changes, and not when `i` changes.

ACTION: Add the signal to the sensitivity list. For example, in the Process Statement above, you must add `i` to the sensitivity list:

```
PROCESS(clk, i)
BEGIN
   o <= i;
END PROCESS;
```

See also:

Section 9.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Qualified Expression error at <*location*>: <*type*> type specified in Qualified Expression must match <*type*> type that is implied for expression by context

---

CAUSE: In a Qualified Expression at the specified location in a VHDL Design File (**.vhd**), you specified a type for an expression. However, the type you specified does not match the type that is implied for the expression based on where the expression occurs (the text of the message indicates the two different types). The type you specify for an expression in a Qualified Expression and the type that is implied for the expression must be the same.

ACTION: Change the type you specify for an expression to be the same as the implied expression type, or change the context in which the expression occurs so the implied expression type is the same as the type you specify.

See also:

Section 7.3.4 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Qualified Expression error at <*location*>: type specified for expression must be valid

---

CAUSE: In a Qualified Expression at the specified location in a VHDL Design File (**.vhd**), you specified a type for an expression. However, the type you specified is not a valid type. For example, you may have specified the alias type for an expression, but an alias type is not a valid type.

ACTION: In the Qualified Expression, specify a valid type for the expression.

See also:

Section 7.3.4 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Resolution Function error at <*location*>: return value or input parameter elements of Resolution Function <*name*> have <*type*> type, but must have same type as resolved signal

---

CAUSE 1:	In a VHDL Design File (**.vhd**) at the specified location, you specified a Resolution Function for a signal. However, the specified type of the Resolution Function's return value is different from the type of the resolved signal. The Resolution Function's return value must have the same type as the signal to which the Resolution Function is associated.
ACTION:	Change the return value so it has the same type as the resolved signal.
CAUSE 2:	In a VHDL Design File (**.vhd**) at the specified location, you specified a Resolution Function for a signal. However, the elements in the Resolution Function's input parameter (which is a one-dimensional, unconstrained array) have the specified type, which is different from the type of the resolved signal. The Resolution Function's input parameter elements must have the same type as the signal to which the Resolution Function is associated.
ACTION:	Change the elements in the input parameter so they have the same type as the resolved signal.

See also:

Sections 2.4, 3, and 4.3.1.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Resolution Function error at <*location*>: signal with access type or file type cannot have Resolution Function

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you specified a Resolution Function for a signal. However, you declared the signal with an access type or file type in a Signal Declaration. You cannot assign a Resolution Function to a signal with access type or file type.

ACTION: Remove the Resolution Function from the signal or declare the signal so it does not have an access type or file type.

See also:

Sections 2.4 and 4.3.1.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Return Statement error at <*location*>: Return Statement for function in Subprogram Declaration must have expression

---

CAUSE:	In a Subprogram Declaration for a function at the specified location in a VHDL Design File (**.vhd**), you specified a Return Statement that does not have an expression. However, the Return Statement for a function must have an expression.
ACTION:	Add an expression to the Return Statement.

See also:

Sections 2.1 and 8.12 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Return Statement error at <*location*>: Return Statement for procedure in Subprogram Declaration cannot have expression

---

CAUSE:	In a Subprogram Declaration for a procedure at the specified location in a VHDL Design File (**.vhd**), you specified a Return Statement that has an expression. However, the Return Statement for a procedure cannot have an expression.
ACTION:	Remove the expression from the Return Statement.

See also:

Sections 2.1 and 8.12 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Return Statement error at <*location*>: Return Statement must be in Subprogram Declaration

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used a Return Statement. However, the Return Statement is not in a Subprogram Declaration. You must use a Return Statement only in a Subprogram Declaration.

ACTION: Remove the Return Statement, or place the Return Statement in a Subprogram Declaration.

See also:

Sections 2.1 and 8.12 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Selected Signal Assignment warning at <*location*>: Selected Signal Assignment choices do not cover all possible values of expression

---

CAUSE: In a Selected Signal Assignment at the specified location in a VHDL Design File (**.vhd**), you specified choices for a Selected Signal Assignment expression. However, the choices do not cover all possible values of the expression; as a result, errors may occur during the future processing of the design.

ACTION: Add choices for all possible values of the expression, or add an `OTHERS` choice, which covers all possible values that are not included in the other Selected Signal Assignment choices.

See also:

Section 9.5.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Signal Assignment Statement error at <*location*>: guarded Signal Assignment Statement must be in guarded Block Statement

---

CAUSE:	In a VHDL Design File (**.vhd**) at the specified location, you used a guarded Signal Assignment Statement outside of a guarded Block Statement. However, you must use guarded Signal Assignment Statements only in guarded Block Statements.
ACTION:	Remove the guarded Signal Assignment Statement, or move the guarded Signal Assignment Statement into a guarded Block Statement.

See also:

Sections 5.3 and 9.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Signal Assignment Statement error at <*location*>: Signal Assignment Statement must use <= to assign value to signal <*name*>

CAUSE:	In a Signal Assignment Statement at the specified location in a VHDL Design File (**.vhd**), you used something other than <= to assign a value to the specified signal. However, you must use <= to assign a value to a signal.
ACTION:	Make sure you use <= in the Signal Assignment Statement.

See also:

Section 8.4 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Signal Assignment Statement warning at <*location*>: ignored all but the first waveform in Signal Assignment Statement

---

CAUSE:	In a Signal Assignment Statement at the specified location in a VHDL Design File (**.vhd**), you used more than one waveform. However, a Signal Assignment Statement can contain only one waveform. As a result, Integrated Synthesis ignored all but the first waveform in the Signal Assignment Statement.
ACTION:	If you do not want Integrated Synthesis to use only the first waveform for the Signal Assignment Statement, or to avoid receiving this message in the future, change the Signal Assignment Statement to a construct that can contain more than one waveform. Otherwise, no action is required.

See also:

Section 8.4 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Signal Declaration error at <*location*>: guarded signal with scalar type must be a resolved signal or a subelement of a resolved signal

---

CAUSE: In a Signal Declaration at the specified location in a VHDL Design File (**.vhd**), you declared a guarded signal that has a scalar type, and that is not a resolved signal or a subelement of a resolved signal. However, a guarded signal with scalar type must be a resolved signal or a subelement of a resolved signal.

ACTION: Change the Signal Declaration so the guarded signal with scalar type is a resolved signal or a subelement of a resolved signal.

See also:

Section 4.3.1.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Signal Declaration error at <*location*>: signal cannot have file type or access type

---

CAUSE: In a Signal Declaration at the specified location in a VHDL Design File (**.vhd**), you declared a signal that has a file type or access type. However, a signal cannot have a file or access type.

ACTION: Change the Signal Declaration so it does not specify a file or access type for the signal, or change the Type Declaration for the type in the Signal Declaration so the type is not a file or access type.

See also:

Sections 4.1 and 4.3.1.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Signal Declaration error at <*location*>: signal must have defined range

---

CAUSE: In a Signal Declaration at the specified location in a VHDL Design File (**.vhd**), you declared a signal. However, you used the undefined range symbol (<>) for the signal. A signal must have a defined range.

ACTION: Define a range for the signal.

See also:

Sections 3.2.1 and 4.3.1.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Signal Declaration warning at <*location*>: ignored initial value specified for signal in Signal Declaration

---

CAUSE: In a Signal Declaration at the specified location in a VHDL Design File (**.vhd**), you specified an initial value for a signal. However, because the initialization of the signal occurred before the start of a process in which the signal is assigned or modified, the MAX+PLUS II Advanced Synthesis software ignored the initial value that you specified in the Signal Declaration.

ACTION: To avoid receiving this message in the future, make sure the signal is assigned an initial value after the execution of the Signal Declaration in the process, and remove the initial value from the Signal Declaration. Otherwise, no action is required.

See also:

Section 4.3.1.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Signal or Variable Assignment Statement error at <*location*>: value assigned to signal or variable must belong to signal or variable range

---

CAUSE: In a Signal Assignment Statement or Variable Assignment Statement at the specified location in a VHDL Design File (**.vhd**), you assigned a value to a signal or variable. However, the value you assigned does not belong to the range of the signal or variable, which you specified in a Signal Declaration or Variable Declaration. The value you assign to a signal or variable must belong to the signal or variable range.

ACTION: Change the value so it belongs to the signal or variable range, or change the signal or variable range to include the value.

See also:

Section 4.3.1.2, 4.3.1.3, 8.4, and 8.5 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Subprogram Body error at <*location*>: function <*name*> does not return a value for all possible conditions

---

CAUSE: In a Subprogram Body at the specified location in a VHDL Design File (**.vhd**), you specified conditions for the return value of the specified function. However, the Subprogram Body does not cover all the possible conditions for the function. For example, in the following code, the function returns a value for `s = '0'`, but not for `s = '1'`.

```
ENTITY example IS
   PORT
   (
      a : IN     BIT;
      b : IN     BIT;
      o : OUT  BIT
   );
END example;
ARCHITECTURE a OF example IS
   FUNCTION exa (s,r : BIT) RETURN BIT IS
   BEGIN
      IF s = '0' THEN
         RETURN (s and r);
      END IF;
   END exa;
BEGIN
      o <= exa (s=>a, r=>b);
END a;
```

ACTION: Make sure the function can return a value for all possible conditions.

See also:

Section 2.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Subprogram Body error at <*location*>: Subprogram Body cannot contain Signal Declaration

---

CAUSE: In a Subprogram Body at the specified location in a VHDL Design File (**.vhd**), you used a Signal Declaration. However, you cannot use a Signal Declaration in a Subprogram Body.

ACTION: Remove the Signal Declaration from the Subprogram Body, or place the Signal Declaration outside the Subprogram Body.

See also:

Sections 2.2 and 4.3.1.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Subprogram Body error at <*location*>: Subprogram Body cannot contain Wait Statement

---

CAUSE: In a Subprogram Body at the specified location in a VHDL Design File (**.vhd**), you used a Wait Statement. However, you cannot use a Wait Statement in a Subprogram Body.

ACTION: Remove the Wait Statement from the Subprogram Body, or place the Wait Statement outside the Subprogram Body.

See also:

Sections 2.2 and 8.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Subprogram Declaration error at <*location*>: declaration of function or procedure <*name*> must have corresponding Subprogram Body

---

CAUSE: In a Subprogram Declaration at the specified location in a VHDL Design File (**.vhd**), you declared the specified function or procedure. However, you did not specify a Subprogram Body that describes the execution of the function or procedure. A Subprogram Declaration must have a corresponding Subprogram Body.

ACTION: Add a Subprogram Body for the Subprogram Declaration. If a Subprogram Body already exists for the Subprogram Declaration, make sure the specification and declarative information in the Subprogram Body matches the information in the Subprogram Declaration.

See also:

Sections 2.1 and 2.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Subprogram Declaration error at <*location*>: mode for formal parameter cannot be <*name*>

CAUSE: In a Subprogram Declaration at the specified location in a VHDL Design File (**.vhd**), you used the specified mode for a formal parameter. However, the formal parameter must have a mode of `IN`, `INOUT`, or `OUT`.

ACTION: Change the mode of the formal parameter to `IN`, `INOUT`, or `OUT`.

See also:

Section 2.1.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Subprogram Declaration error at <*location*>: mode for formal parameter of function cannot be <*name*>

CAUSE: In a Subprogram Declaration at the specified location in a VHDL Design File (**.vhd**), you used the specified mode for a formal parameter of a function. However, the formal parameter must have a mode of `IN`.

ACTION: Change the mode of the formal parameter to `IN`.

See also:

Section 2.1.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Subprogram Declaration error at <*location*>: object class for formal parameter cannot be <*name*>

CAUSE: In a Subprogram Declaration at the specified location in a VHDL Design File (**.vhd**), you used the specified object class for a formal parameter. However, the formal parameter must have an object class of `CONSTANT`, `VARIABLE`, `SIGNAL`, or `FILE`.

ACTION: Change the object class of the formal parameter to `CONSTANT`, `VARIABLE`, `SIGNAL`, or `FILE`.

See also:

Sections 2.1.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Subprogram Declaration error at <*location*>: object class for formal parameter of function cannot be <*name*>

CAUSE: In a Subprogram Declaration at the specified location in a VHDL Design File (**.vhd**), you used the specified object class for a formal parameter of a function. However, the formal parameter must have an object class of `CONSTANT`, `SIGNAL`, or `FILE`.

ACTION: Change the mode of the formal parameter to `CONSTANT`, `SIGNAL`, or `FILE`.

See also:

Sections 2.1.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Subtype Declaration error at <*location*>: subtype for constrained <*type*> type cannot have range

---

CAUSE: In a Subtype Declaration at the specified location in a VHDL Design File (**.vhd**), you declared a subtype for the specified array type. However, both the subtype and the type are constrained, that is, both have a range. If a type has a range, the subtype that is based on the type cannot have a range.

ACTION: Remove the range from the subtype or type.

See also:

Sections 3.1, 4.1, and 4.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Subtype Declaration error at <*location*>: subtype range must belong to range for <*type*> type

---

CAUSE: In a Subtype Declaration at the specified location in a VHDL Design File (**.vhd**), you declared a subtype for the specified type. However, you specified a range for the subtype that exceeds the bounds of the range that you specified for the type in a Type Declaration. The range for the subtype must be a subset of the range for the type on which the subtype is based.

ACTION: Change the range for the subtype or the type so that the range for the subtype is a subset of the range for the type.

See also:

Sections 4.1 and 4.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Subtype or Type Declaration error at <*location*>: bounds in range must both have integer type or floating point value type

---

CAUSE: In a Subtype Declaration or Type Declaration at the specified location in a VHDL Design File (**.vhd**), you declared a range for a subtype or type. However, at least one of the bounds of the range does not have an integer type or a floating point value type. The bounds of the range must both have an integer type, or must both have a floating point value type.

ACTION: Change one or both bounds in the subtype or type range so they both have an integer type or a floating point value type.

See also:

Sections 3.1, 4.1, and 4.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Subtype or Type Declaration warning at <*location*>: subtype or type has null range. Switching left bound and right bound of range.

CAUSE: In a Subtype Declaration or Type Declaration at the specified location in a VHDL Design File (**.vhd**), you declared a range for a subtype or type. However, you either specified an ascending range and specified a left bound for the range that is larger than the range's right bound, or you specified a descending range and specified a right bound for the range that is larger than the range's left bound. As a result, the range is empty, or null. Integrated Synthesis switched the left bound and right bound so that the range contains a valid set of indices.

ACTION: To avoid receiving this message in the future, switch the left and right bounds of the range so that an ascending range has a right bound that is larger than the left bound, or so that a descending range has a left bound that is larger than the right bound. Otherwise, no action is required.

See also:

Sections 4.1 and 4.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL syntax error at <*location*>: name used in construct must match previously specified name <*name*>

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used a name in a construct. However, the name you used does not match the name you previously specified for the construct. For example, the Architecture Body in the following code contains the name `ex2` after the `END` keyword, but the name of the Architecture Body is specified as `ex1`:

```
ARCHITECTURE ex1 OF example IS
...
END ex2;
```

You must use the same name throughout the construct.

ACTION: Make sure you use the same name throughout the construct. In the example above, you must change the name after the `END` keyword to `ex1`, or change the name you specified for the Architecture Body to `ex2`.

See also:

Section 6 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL syntax error at <*location*>: object with <*type*> type cannot contain character <*character*>'

CAUSE: In a Constant Declaration, Signal Declaration, Variable Declaration, Signal Assignment Statement, or Variable Assignment Statement at the specified location in a VHDL Design File (**.vhd**), you assigned the specified character to an object. You also specified a type for the object in a Constant Declaration, Signal Declaration, or Variable Declaration. However, an object with the specified type cannot contain the character.

ACTION: Make sure to assign only legal characters to the object, or change the type of the object so it can contain the character you assigned.

See also:

Sections 3, 4.3.1.2, 4.3.1.3, 8.4, and 8.5 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL syntax error at <*location*>: right bound of range must be a constant

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you specified a range. However, the right bound of the range you specified is not a constant. The right bound of a range must be a constant.

ACTION: Change the right bound of the range to a constant.

See also:

Section 3.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL syntax error at <*location*>: syntax error occurred at or near text <*text*>

CAUSE: In a VHDL Design File (**.vhd**), a syntax error occurred at or near the specified text. For example, a keyword is misspelled or required punctuation (such as a semicolon or parenthesis) is missing.

ACTION: Check for syntax errors in the VHDL Design File. Refer to the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual* for information on the correct syntax for VHDL constructs.

VHDL syntax error at <*location*>: syntax error occurred near text <*text*>

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, a syntax error occurred near the specified text. For example, a keyword is misspelled or required punctuation (such as a semicolon or parenthesis) is missing.

ACTION: Check for syntax errors in the VHDL Design File. Refer to the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual* for information on the correct syntax for VHDL constructs.

VHDL syntax error: experienced unexpected end-of-file -- delimiter or keyword may be missing

---

CAUSE:	In a VHDL Design File (**.vhd**), Integrated Synthesis experienced an unexpected end-of-file.
ACTION:	Make sure you did not omit semicolons, keywords, or other necessary delimiters or text from the VHDL Design File.

See also:

The IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Type Conversion error at <*location*>: <*type*> type cannot be converted to <*type*> type

---

CAUSE: In a Type Conversion at the specified location in a VHDL Design File (**.vhd**), you attempted to convert the specified type of an object to a new type (which is also specified in the text of the message). However, you cannot convert the object type to the new type. For example, you cannot use a Type Conversion to convert the bit type of an object to real type.

ACTION: Remove the Type Conversion, or change the Type Conversion so it converts the object type to a valid new type. Or, create a function that explicitly performs the desired type conversion.

See also:

Section 7.3.5 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Type Conversion error at <*location*>: can't determine type of object or expression near text or symbol <*text*>

CAUSE: In a Type Conversion at the specified location and near the specified text or symbol in a VHDL Design File (**.vhd**), you attempted to convert the type of an object or expression. However, Integrated Synthesis cannot determine the original type of the object or expression because the object or expression has more than one possible type.

ACTION: Make sure the design clearly specifies the original type of the object or expression. For example, you can explicitly specify a type for an object using a qualified expression.

See also:

Sections 7.3.4 and 7.3.5 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Type Conversion error at <*location*>: converted type of object near text or symbol <*text*> must match <*type*> type of target object

---

CAUSE: In a Type Conversion at the specified location and near the specified text or symbol in a VHDL Design File (**.vhd**), you converted the type of an object. You also assigned the object with the converted type to a target object. However, the converted type of the object does not match the specified type of the target object. The converted type of the object and the type of the target object must be the same.

ACTION: Change the type to which you are converting or change the type of the target object so both types are the same.

See also:

Section 7.3.5 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Type Conversion error at <*location*>: type <*type*> of formal parameter in positional Association List cannot be converted to actual parameter type <*type*>

CAUSE: In a Type Conversion at the specified location in a VHDL Design File (**.vhd**), you attempted to convert the specified type of a formal parameter in a positional Association List to the specified type of the actual parameter you associated with the formal parameter. However, you cannot convert the formal parameter type to be the same as the actual parameter type. For example, the Type Conversion function may not exist, or the types may not be closely related to allow a Type Conversion.

ACTION: Make sure you can convert the formal parameter type to the actual parameter type. For example, implement a Type Conversion function, make sure the types of the formal parameter and actual parameter are closely related, or change the actual parameter type to be the same as the formal parameter.

See also:

Sections 4.3.2.2 and 7.3.5 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Type Conversion error at <*location*>: Type Conversion near text or symbol <*text*> must have one argument

---

CAUSE: In a Type Conversion at the specified location and near the specified text or symbol in a VHDL Design File (**.vhd**), you attempted to convert the type of an object. However, you either did not specify an argument for the Type Conversion, or you specified more than one argument for the Type Conversion. A Type Conversion must have only one argument that specifies the object on which you want to perform the conversion.

ACTION: Make sure the Type Conversion has one argument. If the Type Conversion has more than one argument, specify a separate Type Conversion for each object on which you want to perform the conversion.

See also:

Section 7.3.5 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Type Declaration error at <*location*>: array type has index range of <*type*> type, but must have index range of discrete type

---

CAUSE: In a Type Declaration at the specified location in a VHDL Design File (**.vhd**), you specified a constrained or unconstrained index range for an array type. However, the index range has the specified floating type, which you cannot use to specify an index range. Instead, the index must have a discrete type, for example, an integer type or an enumeration type.

ACTION: Change the index range type to a discrete type.

See also:

Sections 3.2.1 and 4.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Type Declaration error at <*location*>: constrained array type contains unconstrained element type, but must contain constrained element type

---

CAUSE:	In a Type Declaration at the specified location in a VHDL Design File (**.vhd**), you used an unconstrained element type in a constrained array type. However, an element type in a constrained array type must be constrained.
ACTION:	Change the element type so it is constrained, or change the array type so it is unconstrained.

See also:

Sections 3.2.1 and 4.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Type Declaration error at <*location*>: element type for array type cannot be unconstrained

---

CAUSE: In a Type Declaration at the specified location in a VHDL Design File (**.vhd**), you specified an element type for an unconstrained array type. However, the element type you specified is an unconstrained array type. You cannot use an unconstrained array type for elements in an unconstrained array type. For example, if you define `ex1` as an unconstrained array type using the following Type Declaration:

```
TYPE ex1 IS ARRAY (NATURAL RANGE<>) OF BIT;
```

You cannot use the following Type Declaration, which specifies `ex1` as the element type for the unconstrained array type `ex2`:

```
TYPE ex2 IS ARRAY (NATURAL RANGE<>) OF ex1;
```

ACTION: Add a range to the array type that you are specifying for the unconstrained array type's elements, or specify a different element type. In the example above, you must add a range to the Type Declaration for `ex1`, or replace `ex1` in the Type Declaration for `ex2` with a type that is not an unconstrained array type.

See also:

Sections 3.2.1 and 4.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Type Declaration error at <*location*>: illegal constrained element in unconstrained array declaration

---

CAUSE: In a Type Declaration at the specified location in a VHDL Design File (**.vhd**), you used a constrained element type in an unconstrained array type. However, an element type in a unconstrained array type must be unconstrained.

ACTION: Change the element type so it is unconstrained, or change the array type so it is constrained.

See also:

Sections 3.2.1 and 4.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Type Declaration error at <*location*>: object <*name*> with enumeration type cannot contain enumeration value <*name*> more than once

---

CAUSE:	In a Type Declaration at the specified location in a VHDL Design File (**.vhd**), you listed the enumeration values for the specified object. However, you listed the specified value more than once. An enumeration type object can contain an enumeration value only once.
ACTION:	Make sure that each enumeration value for the enumeration type object is listed only once in the Type Declaration.

See also:

Sections 3.1.1 and 4.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Type Declaration error at <*location*>: range constraint bounds for physical type must have integer type

---

CAUSE: In a Type Declaration at the specified location in a VHDL Design File (**.vhd**), you declared a physical type. However, one or both bounds of the physical type's range constraint does not have an integer type. Both bounds must have an integer type.

ACTION: Change the range constraint bounds so they both have an integer type.

See also:

Sections 3.1.3 and 4.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Type Declaration error at <*location*>: record element cannot have an unconstrained array type

---

CAUSE: In a Type Declaration at the specified location in a VHDL Design File (**.vhd**), you declared a record type. However, you specified a record type element that has an unconstrained array type. An element for a record type cannot have an unconstrained array type.

ACTION: Add a discrete range to the array type that you are specifying for the record element, or specify a record element type that is not an unconstrained array type.

See also:

Sections 3.2.1, 3.2.2, and 4.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Type Declaration error at <*location*>: unit used in secondary unit declaration must be a physical type

---

CAUSE: In a Type Declaration at the specified location in a VHDL Design File (**.vhd**), you used a unit in a secondary unit declaration. However, the unit is not a physical type. The unit in a secondary unit declaration must be a physical type.

ACTION: Change the unit in the secondary unit declaration to a physical type.

See also:

Sections 3.1.3 and 4.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Type Declaration error at <*location*>: value for physical type must have an integer or real type

---

CAUSE: In a Type Declaration at the specified location in a VHDL Design File (**.vhd**), you used a value that does not have an integer or real type to declare a physical type. The value must have an integer or real type.

ACTION: Change the value for the physical type so it has an integer or real type.

See also:

Sections 3.1.3 and 4.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL type mismatch at <*location*>: object(s) associated with operator <*name*> must have <*type*> type

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the specified operator for one or more objects. However, the object(s) do not have the specified type, which is compatible with the operator and with other associated types in the design.

ACTION: Change the type of the object(s), or replace the operator with one that is compatible with the current type of the object(s).

See also:

Section 7.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL type mismatch error at <*location*>: <*type*> type does not match integer literal

---

CAUSE: In a Signal Assignment Statement, Variable Assignment Statement, Subtype Declaration, or Type Declaration at the specified location in a VHDL Design File (**.vhd**), you used an integer literal for the specified type. However, you cannot use integer literals for the specified type.

ACTION: Remove the integer literals from the assignment or declaration, or change the type to one that allows integer literals.

See also:

Sections 3, 4.1, 4.2, 8.4, and 8.5 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL type mismatch error at <*location*>: <*type*> type does not match real literal

---

CAUSE: In a Signal Assignment Statement, Variable Assignment Statement, Subtype Declaration, or Type Declaration at the specified location in a VHDL Design File (**.vhd**), you used a real literal for the specified type. However, you cannot use real literals for the specified type.

ACTION: Remove the real literals from the assignment or declaration, or change the type to one that allows real literals.

See also:

Sections 3, 4.1, 4.2, 8.4, and 8.5 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL type mismatch error at <*location*>: <*type*> type does not match string literal

---

CAUSE: In a Signal Assignment Statement, Variable Assignment Statement, Subtype Declaration, or Type Declaration at the specified location in a VHDL Design File (**.vhd**), you used a string literal for the specified type. However, you cannot use string literals for the specified type.

ACTION: Remove the string literals from the assignment or declaration, or change the type to one that allows string literals.

See also:

Sections 3, 4.1, 4.2, 8.4, and 8.5 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL type mismatch error at <*location*>: function <*name*> does not accept type <*type*>

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, the specified function does not have the given type associated with it as an argument.

ACTION: Create a new polymorphic version of the specified function, or performa type conversion on the value passed to the function.

See also:

Section 2.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL type mismatch error at <*location*>: type <*type*> used with bit string literal must be BIT type

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the specified type with a bit string literal. However, the type you use with a bit string literal must be a BIT type.

ACTION: Make sure the type you use with the bit string literal is a BIT type.

See also:

Section 3.2.1.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL type mismatch error at <*location*>: type of formal parameter <*name*> does not match <*type*> type of value

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you assigned a value to the specified formal parameter. However, the type of the formal parameter does not match the specified type of the value. The types of the formal parameter and the value must match.

ACTION: Correct the type of either the formal parameter or the value that you assigned to the formal parameter so the two types are the same.

See also:

The IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL type mismatch error at <*location*>: type of indexed object that is assigned or mapped to target object must match target object type <*type*>

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you assigned an indexed object to a target object. However, in the object's declaration, you declared a type for the object that is not the same as the specified type you declared for the target object. The type of the object and the type of the target object must be the same. For example, the Signal Assignment Statement in the following code tries to assign the indexed object `i(0)` with std_logic type to the target object `my_sig` with bit type. However, the types of the two objects must be the same.

```
ENTITY example IS
   PORT
   (
      i :  IN   std_logic_vector (0 to 3);
      o :  OUT  std_logic_vector (0 to 3)
   );
END example;

ARCHITECTURE  ex1 OF example IS
   signal my_sig : bit;
BEGIN
   my_sig <= i(0);
END ex1;
```

ACTION: Change the indexed object or the target object so they have the same type. In the example above, you can change signal `my_sig` so it has the std_logic type.

See also:

Sections 3.2.1 and 4.3.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Type or Variable Declaration error at <*location*>: bounds of type or variable range must have same type

---

CAUSE: In a Type Declaration or Variable Declaration at the specified location in a VHDL Design File (**.vhd**), you specified a range for a type or variable. However, the bounds of the range do not have the same type. The range bounds must have the same type.

ACTION: Change the range bounds types so they are the same.

See also:

Sections 3, 4.1, and 4.3.1.3 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL unsupported feature error at <*location*>: nonobject aliases are not supported

---

CAUSE: In an Alias Declaration at the specified location in a VHDL Design File (**.vhd**), you declared a nonobject alias, that is, an alias for a named entity that is not an object. However, Integrated Synthesis does not support nonobject aliases.

ACTION: Delete the Alias Declaration for the nonobject alias, or change the Alias Declaration so it declares an alias for an object.

See also:

Section 4.3.3 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Use Clause error at <*location*>: design library <*name*> does not contain primary unit <*name*>

CAUSE: In a Use Clause at the specified location in a VHDL Design File (**.vhd**), you tried to access the specified primary unit of the specified design library. However, the design library does not contain the primary unit, or the primary unit has not been successfully compiled. For example, the Use Clause in the following code attempts to access the primary unit `example`, but the **ieee** design library does not contain the primary unit `example`:

```
LIBRARY ieee;
USE ieee.example;
```

ACTION: Access only primary units that exist in a design library and have been successfully compiled.

See also:

Sections 6.3, 10.2, 10.4, and 11.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Variable Assignment Statement error at <*location*>: target object <*name*> must be variable or aggregate

---

CAUSE: In a Variable Assignment Statement at the specified location in a VHDL Design File (**.vhd**), you used the specified target object, which is not a variable or an aggregate. However, the target object of a Variable Assignment Statement must be either a variable or an aggregate.

ACTION: Change the target object into a variable or an aggregate, or change the Variable Assignment Statement into an appropriate construct (for example, if the target object is a signal, change the Variable Assignment Statement into a Signal Assignment Statement).

See also:

Section 8.5 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Variable Assignment Statement error at <*location*>: Variable Assignment Statement must use := to assign value to variable <*name*>

CAUSE: In a Variable Assignment Statement at the specified location in a VHDL Design File (**.vhd**), you used something other than := to assign a value to the specified variable. However, you must use := to assign a value to a variable.

ACTION: Make sure you use := in the Variable Assignment Statement.

See also:

Section 8.5 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Variable Declaration error at <*location*>: Variable Declaration cannot have a file subtype indication

---

CAUSE: In a Variable Declaration at the specified location in a VHDL Design File (**.vhd**), you used a file subtype indication, However, you cannot use a file subtype indication in a Variable Declaration.

ACTION: Change the Variable Declaration to a File Declaration.

See also:

Sections 4.3.1.3 and 4.3.1.4 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Variable Declaration error at <*location*>: variable declared in subprogram or process cannot be a shared variable

---

CAUSE: In a Variable Declaration at the specified location in a VHDL Design File (**.vhd**), you declared a shared variable in a subprogram or process. However, a variable you declare in a subprogram or process cannot be a shared variable.

ACTION: Make sure that the Variable Declaration does not contain the keyword `shared`, that is, does not declare the variable as a shared variable.

See also:

Section 4.3.1.3 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Variable Declaration error at <*location*>: variable declared outside of subprogram or process must be a shared variable

---

CAUSE: In a Variable Declaration at the specified location in a VHDL Design File (**.vhd**), you declared a variable that is not shared. However, you declared the variable outside of a subprogram or process. A variable you declare outside of a subprogram or process must be a shared variable.

ACTION: Make sure that the Variable Declaration contains the keyword `shared`, that is, declares the variable as a shared variable. Or, place the variable inside a subprogram or process.

See also:

Section 4.3.1.3 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Variable Declaration warning at <*location*>: ignored initial value specified for variable in Variable Declaration.

CAUSE: In a Variable Declaration at the specified location in a VHDL Design File (**.vhd**), you specified an initial value for a variable. However, because the initialization of the variable occurred before the start of a process in which the variable is assigned or modified, the MAX+PLUS II Advanced Synthesis software ignored the initial value that you specified in the Variable Declaration.

ACTION: To avoid receiving this message in the future, make sure the variable is assigned an initial value after the execution of the Variable Declaration in the process, and remove the initial value from the Variable Declaration. Otherwise, no action is required.

See also:

Section 4.3.1.3 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Wait Statement error at <*location*>: condition clause with UNTIL keyword cannot contain contain 'EVENT predefined attribute

---

CAUSE: In a Wait Statement at the specified location in a VHDL Design File (**.vhd**), you used the `'EVENT` predefined attribute in a condition clause that has the `UNTIL` keyword. For example, you may have used `clk'EVENT` in the condition clause. However, a condition clause cannot contain the `'EVENT` predefined attribute.

ACTION: Make sure the condition clause with the `UNTIL` keyword in the Wait Statement does not contain the `'EVENT` predefined attribute.

See also:

Sections 8.1 and 14.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL Wait Statement error at <*location*>: Wait Statement must contain condition clause with UNTIL keyword

---

CAUSE: In a Wait Statement at the specified location in a VHDL Design File (**.vhd**), you did not use a condition clause with the `UNTIL` keyword.

ACTION: Change the Wait Statement so it contains a condition clause with the `UNTIL` keyword.

See also:

Section 8.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL warning at <*location*>: ENUM_ENCODING attribute specifies too many encoding values for enumeration type <*type*>

CAUSE: In an Attribute Specification at the specified location in a VHDL Design File (**.vhd**), you specified the `ENUM_ENCODING` attribute for an object with the specified enumeration type, which you declared for a state machine. However, the expression in the Attribute Specification contains more encoding values than there are enumeration literals in the enumeration type. The number of encoding values in the Attribute Specification must match the number of enumeration literals you declared for the enumeration type in a Type Declaration.

ACTION: Make sure the Attribute Specification contains the correct number of encoding values.

☞ The `ENUM_ENCODING` attribute is MAX+PLUS II-specific, and may not be available with other VHDL tools.

See also:

Manually Specifying State Assignments
Sections 3.1.1, 4.1, and 5.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL warning at <*location*>: Exemplar attribute or directive <*name*> is not supported in MAX+PLUS II Advanced Synthesis software version 2.1 and earlier -- only translate_off and translate_on are supported

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the specified Exemplar attribute or directive. However, the MAX+PLUS II Advanced Synthesis software version 2.1 or earlier supports only the `translate_off` and `translate_on` directives.

ACTION: To avoid problems in the future processing of your design, or to avoid receiving this message in the future, remove the unsupported attribute or directive.

VHDL warning at <*location*>: ignored assignment of value to null range

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you assigned a value to a null range. Because you cannot make an assignment to a null range, the MAX+PLUS II Advanced Synthesis software ignored the assignment.

ACTION: Make sure the range should be a null range. If the range is suppose to be a null range, and you want to avoid receiving this message in the future, remove the assignment to the null range.

See also:

Section 3.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL warning at <*location*>: ignored choice that is null range

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used a choice that is a null range. A choice cannot be a null range; therefore, Integrated Synthesis ignored the choice.

ACTION: If you do not want Integrated Synthesis to ignore the choice, or to avoid receiving this message in the future, change the choice so it is not a null range. Otherwise, no action is required.

See also:

Sections 3.1, 7.3.2, 8.8, and 9.5.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL warning at <*location*>: ignored choice with illegal range bounds

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used a choice that contains range bounds. However, the range bounds are illegal; therefore, Integrated Synthesis ignored the choice.

ACTION: If you do not want Integrated Synthesis to ignore the choice, or to avoid receiving this message in the future, make sure the range bounds in the choice are legal. Otherwise, no action is required.

See also:

Sections 7.3.2 and 8.8 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL warning at <*location*>: ignored choice with meta-value <*name*> for synthesis

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used a choice that contains the specified meta-value (such as X or Z). For synthesis, the choice cannot contain the meta-value; therefore, Integrated Synthesis ignored the choice.

ACTION: If you do not want Integrated Synthesis to ignore the choice, or to avoid receiving this message in the future, change the choice so it does not contain meta-values. Otherwise, no action is required.

See also:

Sections 7.3.2, 8.8, and 9.5.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL warning at <*location*>: ignored choice with meta-values

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used a choice that contains one or more meta-values (such as X or Z). A choice cannot contain meta-values; therefore, Integrated Synthesis ignored the choice.

ACTION: If you do not want Integrated Synthesis to ignore the choice, or to avoid receiving this message in the future, change the choice so it does not contain meta-values. Otherwise, no action is required.

See also:

Sections 7.3.2, 8.8, and 9.5.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL warning at <*location*>: ignored pragma <*name*> for constant net

---

CAUSE: In a VHDL Design File (**.vhd**), you applied the specified pragma to nets. Integrated Synthesis cannot use the pragma for constant nets, and therefore it ignored the pragma for the constant net at the specified location.

ACTION: Make sure that applying the pragma to a constant net is producing the desired behavior. To avoid receiving this message in the future, make sure you do not use the pragma for a constant net. Otherwise, no action is required.

VHDL warning at <*location*>: ignored synthesis directive attribute <*name*> because number of bits in <*type*> type does not match number of enumeration values in Type Declaration

---

CAUSE:	In a VHDL Design File (**.vhd**) at the specified location, you used the specified synthesis directive attribute. However, the number of bit values for the specified type (which is related to the synthesis directive attribute) is not the same as the number of enumeration values in the Type Declaration for the type. As a result, Integrated Synthesis ignored the synthesis directive attribute.
ACTION:	If you do not want Integrated Synthesis to ignore the synthesis directive attribute, or to avoid receiving this message in the future, make sure the number of bit values in the type matches the number of enumeration values in the type's Type Declaration. Otherwise, no action is required.

See also:

Sections 4.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL warning at <*location*>: ignored unknown or unsupported pragma <*name*>

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, the specified text, which appears to be a VHDL pragma, is not recognized by the MAX+PLUS II Advanced Synthesis software. It could be that the text is a valid third-party pragma that is not supported in the MAX+PLUS II Advanced Synthesis software, or the text could be incorrectly typed.

ACTION: Remove or correct the offending pragma.

VHDL warning at <*location*>: ignored VHDL standard library NOW function, which is not supported for synthesis

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used the `NOW` function of the VHDL **standard** library, which is not supported for synthesis. As a result, Integrated Synthesis ignored the function.

ACTION: To avoid receiving this message in the future, make sure the VHDL Design File does not contain the `NOW` function. Otherwise, no action is required.

See also:

Section 11.2 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL warning at <*location*>: used initial value for variable <*name*> because it was never assigned

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you made an assignment using a variable that was never assigned.

ACTION: Assign the variable to a known value, and then use the variable to assign to another variable.

See also:

Sections 4.2.1.3 and 8.5 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

VHDL warning at <*location*>: used X for unrecognized character <*name*> in enumerated type

---

CAUSE: In a VHDL Design File (**.vhd**) at the specified location, you used an enumerated type with the specified character. However, Integrated Synthesis does not recognize the character. As a result, Integrated Synthesis used X for the character.

ACTION: If you do not want Integrated Synthesis to use X for the unrecognized character, or to avoid receiving this message in the future, use only recognizable characters in the enumerated type. Otherwise, no action is required.

See also:

Section 3.1.1 of the IEEE Std 1076-1993 *IEEE Standard VHDL Language Reference Manual*

Virtual pin clock <*name*> for virtual pin <*name*> does not exist in the design -- auto-selecting clock

---

CAUSE: You assigned the **Virtual Pin Clock** logic option to the specified pin. However this pin does not exist in the current design. When this condition occurs, the Compiler automatically selects an appropriate virtual pin clock.

ACTION: No action is required. To avoid receiving this message in the future, assign the **Virtual Pin Clock** assignment to an existing clock signal in the design.

Virtual pin clock not found for <*number*> virtual pins -- used Auto clock

---

CAUSE: You did not specify a valid **Virtual Pin Clock** assignment for the specified number of Virtual Pins in the design. As a result, the Compiler converted the virtual pins in question to registered lcells, and automatically detects the clock associated with the virtual pin.

ACTION: No action is required. To avoid receiving this message in the future, assign the **Virtual Pin Clock** assignment to an existing clock signal in the design.

Virtual pins cannot be assigned to ports requiring real pins

CAUSE: You assigned the **Virtual Pin** logic option to one or more ports that require real pins. The **Virtual Pin** logic option is illegal for ports that require a connection to real pins.

ACTION: Remove the **Virtual Pin** assignment from any ports requiring connection to a real pin.

waddr port is not connected in WYSIWYG RAM primitive *<name>*

CAUSE: The specified WYSIWYG RAM primitive does not have a connected `waddr` port. The `waddr` port must be connected.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

When in its current operating mode, WYSIWYG RAM primitive <*name*> cannot have <*name*> parameter

---

CAUSE: The specified WYSIWYG RAM primitive, when in its current operating mode, has the specified parameter. The WYSIWYG RAM primitive cannot have the specified parameter when in its current operating mode. The parameter must be removed.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG CAM primitive <*name*> cannot have matchfound port connected when in an unencoded mode or a multiple match mode

---

CAUSE: The specified WYSIWYG CAM primitive has a `matchfound` port connected when in an unencoded mode or a multiple match mode. The WYSIWYG CAM primitive cannot have a `matchfound` port connected when the WYSIWYG CAM primitive is in an unencoded mode or a multiple match mode. The `matchfound` port must be removed or the content-addressable memory (CAM) mode must be changed.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG CAM primitive <*name*> has actual ADDRESS_WIDTH <*number*>, which is different from declared ADDRESS_WIDTH <*number*>

CAUSE: The specified WYSIWYG CAM primitive has a value for the specified actual `ADDRESS_WIDTH` parameter that is different from the specified declared `ADDRESS_WIDTH` parameter.

ACTION: Check the design file to make sure the parameters and the logic feeding the address ports have the same width value. If the parameters are correct and you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG CAM primitive <*name*> has actual literal width <*number*>, which is different from literal width <*number*> declared in parameter

---

CAUSE: The specified WYSIWYG CAM primitive has a value for the specified actual literal width that is different from the specified literal width value declared in the parameter. The parameters and the logic feeding the literal ports must have the same width value.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG CAM primitive <*name*> must have a datain port

CAUSE: The specified WYSIWYG CAM primitive does not have a `datain` port. The `datain` port must be connected.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG CAM primitive <*name*> must have a lit port

CAUSE: The specified WYSIWYG CAM primitive does not have a `lit` port. The `lit` port must be connected.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG CAM primitive <*name*> must have a waddr port

CAUSE: The specified WYSIWYG CAM primitive does not have a `waddr` port. The `waddr` port must be connected.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG CAM primitive <*name*> must have a we port

CAUSE: The specified WYSIWYG CAM primitive does not have a `we` port. The `we` port must be connected.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG CAM primitive <*name*> must have a wrinvert port

CAUSE: The specified WYSIWYG CAM primitive does not have a `wrinvert` port. The `wrinvert` port must be connected.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG CAM primitive <*name*> must have content-addressable memory ADDRESS_WIDTH parameter

---

CAUSE: The specified WYSIWYG CAM primitive does not have a content-addressable memory (CAM) `ADDRESS_WIDTH` parameter, which specifies the address width of the CAM. A CAM `ADDRESS_WIDTH` parameter with a legal value must be specified.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG CAM primitive <*name*> must have LOGICAL_CAM_DEPTH parameter

---

CAUSE: The specified WYSIWYG CAM primitive does not have a `LOGICAL_CAM_DEPTH` parameter. A `LOGICAL_CAM_DEPTH` parameter with a legal value must be specified.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG CAM primitive <*name*> must have LOGICAL_CAM_WIDTH parameter

---

CAUSE: The specified WYSIWYG CAM primitive does not have a `LOGICAL_CAM_WIDTH` parameter. A `LOGICAL_CAM_WIDTH` parameter with a legal value must be specified.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG CAM primitive <*name*> must have outputselect port connected only when OPERATION_MODE parameter is set to UNENCODED_32_ADDRESS or MULTIPLE_MATCH

CAUSE: The specified WYSIWYG CAM primitive has a connected `outputselect` port, but the primitive's `OPERATION_MODE` parameter is not set to either `UNENCODED_32_ADDRESS` or `MULTIPLE_MATCH`. The `outputselect` port must be removed or the `OPERATION_MODE` parameter value must be changed.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG CAM primitive <*name*> must have outputselect port connected when the OPERATION_MODE parameter is set to UNENCODED_32_ADDRESS or MULTIPLE_MATCH

CAUSE: The specified WYSIWYG CAM primitive does not have a connected `outputselect` port, but the primitive's `OPERATION_MODE` parameter is set to either `UNENCODED_32_ADDRESS` or `MULTIPLE_MATCH`. The `outputselect` port must be connected or the `OPERATION_MODE` parameter value must be changed.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG ClockLock PLL primitive <*name*> cannot be fed by logic

---

CAUSE: The specified WYSIWYG ClockLock PLL primitive is fed by logic, rather than directly by an input pin. However, WYSIWYG ClockLock PLL primitives must be fed directly by an input pin.

ACTION: Correct the design so the WYSIWYG ClockLock PLL primitive is fed directly by a pin.

WYSIWYG CRC block or remote update block primitive *<name>* must use *<name>* port

---

CAUSE: The specified WYSIWYG CRC block or remote update block primitive does not use the specified port. The WYSIWYG CRC block or remote update block primitive must have a connection for the specified port.

ACTION: Edit the design to make sure the specified port is connected properly.

WYSIWYG I/O <*name*> operation mode is <*type*>, which does not match padio pin <*name*> type of <*type*>

CAUSE: The specified WYSIWYG I/O's operation mode does not match the specified type of the specified pin connected to its padio port.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG I/O operation modes do not match their padio pin's type

---

CAUSE: The operation mode of one or more WYSIWYG I/O does not match the type of the pins connected to their padio ports. Additional messages follow that list the specific WYSIWYG name, mode and pins associated with this error.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG I/O primitive <*name*> cannot be converted to an APEX II WYSIWYG I/O primitive

---

CAUSE: The specified WYSIWYG I/O primitive cannot be converted to an APEX II WYSIWYG I/O primitive because the APEX II I/O primitive cannot support the same configuration.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG I/O primitive <*name*> cannot have both a clear signal and power-up high, or both a preset signal and power-up low

---

CAUSE: The specified WYSIWYG I/O primitive has either a clear signal and powers-up high, or has a preset signal and powers-up low. The WYSIWYG I/O primitive cannot have either of these combinations of settings. Either the clear or preset for the corresponding power-up specification must be changed, or the power-up specification must be changed.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG I/O primitive <*name*> cannot have both aclr port and a POWER_UP parameter value set to HIGH

CAUSE: The specified WYSIWYG I/O primitive has both an `aclr` port and a `POWER_UP` parameter value set to `HIGH`. Either the `aclr` port must be removed or the WYSIWYG I/O primitive's `POWER_UP` parameter must be changed to `LOW`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG I/O primitive <*name*> cannot have extra register control signals

---

CAUSE: The specified WYSIWYG I/O primitive has an I/O register that is connected to control signals, but the WYSIWYG I/O primitive does not use the I/O register. The WYSIWYG I/O primitive's register modes must be checked to determine which control signals are connected, and the identified control signals must be removed. Or, the register modes must be added.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications.

WYSIWYG I/O primitive <*name*> cannot have OPEN_DRAIN_OUTPUT parameter value set to TRUE when its I/O element is in bidirectional operation mode and uses an output enable port

---

CAUSE: The specified WYSIWYG I/O primitive has an `OPEN_DRAIN_OUTPUT` parameter value set to `TRUE`, but its I/O element is in bidirectional operation mode and uses an output enable port. The WYSIWYG primitive cannot have an `OPEN_DRAIN_OUTPUT` parameter value set to `TRUE` when its I/O element is in bidirectional operation mode and uses an output enable port. The `OPEN_DRAIN_OUTPUT` parameter value must be set to `FALSE`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG I/O primitive <*name*> cannot specify modes related to unused register

---

CAUSE: The specified WYSIWYG I/O primitive does not use a specific register, but specifies modes related to that register. The register must be added or the register-related modes must be removed.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG I/O primitive <*name*> converted to equivalent logic

CAUSE: The specified WYSIWYG I/O primitive does not have a dedicated top-level pin connected to its `padio` port. When this condition occurs, the MAX+PLUS II Advanced Synthesis software converts the WYSIWYG I/O primitive into equivalent logic.

ACTION: If this conversion is acceptable, no action is required. If this conversion is not acceptable, and if you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG I/O primitive <*name*> has illegal OPERATION_MODE parameter value <*name*>

CAUSE: The specified WYSIWYG I/O primitive has an illegal `OPERATION_MODE` parameter value. The `OPERATION_MODE` parameter value must be changed to one of the following legal values: `INPUT`, `OUTPUT`, or `BIDIR`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG I/O primitive <*name*> has illegal value for FEEDBACK_MODE parameter

---

CAUSE: The specified WYSIWYG I/O primitive has an illegal `FEEDBACK_MODE` parameter value. The illegal parameter value must be replaced with a legal parameter value.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG I/O primitive <*name*> has illegal value for OPEN_DRAIN_OUTPUT parameter

---

CAUSE: The specified WYSIWYG I/O primitive has an illegal `OPEN_DRAIN_OUTPUT` parameter value. The illegal parameter value must be changed to `TRUE` or `FALSE`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG I/O primitive <*name*> has illegal value for REG_SOURCE_MODE parameter

---

CAUSE: The specified WYSIWYG I/O primitive has an illegal `REG_SOURCE_MODE` parameter value. The illegal parameter value must be replaced with a legal parameter value.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG I/O primitive <*name*> has OE_REGISTER_MODE parameter value set to REGISTER, but does not use oe port

---

CAUSE: The specified WYSIWYG I/O primitive has the `OE_REGISTER_MODE` parameter value set to `REGISTER`, but does not use an `oe` port. Either the `oe` port must be added or the WYSIWYG I/O primitive's `OE_REGISTER_MODE` parameter value must be changed.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG I/O primitive <*name*> has OPERATION_MODE parameter value set to BIDIR, but does not have oe port

---

CAUSE: The specified WYSIWYG I/O primitive has the `OPERATION_MODE` parameter value set to `BIDIR`, but does not have an `oe` port. Either the `oe` port must be added or the WYSIWYG I/O primitive's `OPERATION_MODE` parameter value must be changed.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG I/O primitive <*name*> is dependent on unconnected register control signals

---

CAUSE: The specified WYSIWYG I/O primitive uses an I/O register, but the register is not connected to some required control signals. The WYSIWYG I/O primitive must have its I/O register connected to these control signals. The WYSIWYG I/O primitive's register modes must be checked to determine which required control signals are missing, and the identified control signals must be connected to the register. Or, the WYSIWYG I/O primitive's register modes must be removed.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG I/O primitive <*name*> is in unsupported mode

CAUSE: The specified WYSIWYG I/O primitive is in a mode that is not supported by the MAX+PLUS II Advanced Synthesis software.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG I/O primitive <*name*> must have padio port

CAUSE: The specified WYSIWYG I/O primitive does not have a required `padio` port. A `padio` port must be added to the WYSIWYG primitive.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG I/O primitive <*name*> uses its clk port, but does not use its register dataout port

---

CAUSE: The specified WYSIWYG I/O primitive uses its `clk` port, but does not use its registered `dataout` port. The WYSIWYG primitive cannot use its `clk` port without also using its registered `dataout` port. Either a registered `dataout` port must be used or the `clk` port must not be used.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG I/O primitive <*name*>, which has OPERATION_MODE parameter value set to either OUTPUT or BIDIR, must have data source

---

CAUSE: The specified WYSIWYG I/O primitive, which has an `OPERATION_MODE` parameter value set to either `OUTPUT` or `BIDIR`, does not have a data source. The WYSIWYG I/O primitive must have either a `datain` port connection or a `GND` connection, or the WYSIWYG I/O primitive's `REG_SOURCE_MODE` parameter value changed to `PIN_LOOP`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG I/O primitives converted to equivalent logic

CAUSE: The specified WYSIWYG I/O primitives do not have dedicated top-level pins connected to `padio` ports. When this condition occurs, the MAX+PLUS II Advanced Synthesis software converts the WYSIWYG I/O primitives into equivalent logic.

ACTION: If this conversion is acceptable, no action is required. If this conversion is not acceptable, and if you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG I/Os found with illegal padio connections

CAUSE: The Logic Synthesizer found WYSIWYG I/O primitives with illegal `padio` connections. Additional messages list the specific WYSIWYG I/O primitives with illegal connections.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG LCELL primitive <*name*> cannot have cin port when its OPERATION_MODE parameter is set to QFBK_COUNTER

CAUSE: The specified WYSIWYG LCELL primitive has a `cin` port when its `OPERATION_MODE` parameter is set to `QFBK_COUNTER`. The WYSIWYG primitive cannot have a `cin` port when its `OPERATION_MODE` parameter is set to `QFBK_COUNTER`. The `cin` port must be removed from the WYSIWYG primitive.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG LCELL primitive <*name*> cannot have cout port when in normal mode

---

CAUSE: The specified WYSIWYG LCELL primitive has a `cout` port, but the `cout` port must not be used when the WYSIWYG primitive is in normal operating mode (that is, when the `OPERATION_MODE` parameter is set to `NORMAL`). The `cout` port must not be used or must be connected only to the `combout` or `regout` ports.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG LCELL primitive <*name*> cannot have improperly connected cascin port

---

CAUSE: The specified WYSIWYG LCELL primitive has a `cascin` port that is inverted or connected to `VCC`, `GND`, or a pin. However, the `cascin` port must be driven by the `cascout` port of another WYSIWYG LCELL primitive. A WYSIWYG LCELL primitive that uses its `cascout` port must be used in order to send the appropriate signal to the WYSIWYG LCELL primitive.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG LCELL primitive <*name*> cannot have PACKED_MODE parameter set to TRUE when not in normal mode

---

CAUSE: The specified WYSIWYG LCELL primitive, which is not in normal operating mode, has the `PACKED_MODE` parameter set to `TRUE`, but the `PACKED_MODE` parameter is supported only in normal operating mode. The `OPERATION_MODE` parameter must be set to `NORMAL` or the `PACKED_MODE` parameter must be set to `FALSE`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG LCELL primitive <*name*> cannot have sload and/or sclr port(s) when in non-counter modes

---

CAUSE: The specified WYSIWYG LCELL primitive has a `sload` and/or `sclr` port(s), the `OPERATION_MODE` parameter is set to something other than `COUNTER`. The WYSIWYG primitive cannot have `sload` and/or `sclr` port(s) when in non-counter modes. Either the port(s) must be removed from the WYSIWYG primitive, or the `OPERATION_MODE` parameter must be set to `COUNTER`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG LCELL primitive <*name*> cannot have sload and/or sclr port(s) when it also has combout port or is in non-arithmetic modes

---

CAUSE: The specified WYSIWYG LCELL primitive, which has a `combout` port or is in an operating mode other than arithmetic mode, also has an `sload` and/or `sclr` port(s). The WYSIWYG primitive cannot have an `sload` and/or `sclr` port(s) when it also has a `combout` port or is in a non-arithmetic mode. The `sload` and/or `sclr` port(s) must be removed from the WYSIWYG LCELL primitive, the `combout` port must removed, or the operating mode must be changed to arithmetic.

ACTION: If you are using an EDA tool, please contact the technical support for the EDA tool regarding this message. Otherwise, please contact Altera Applications for assistance.

WYSIWYG LCELL primitive <*name*> cannot use both datad and cout ports when in counter mode

---

CAUSE: The specified WYSIWYG LCELL primitive is using the `datad` input when in counter mode. The WYSIWYG LCELL primitive, which is using the `cout` port, cannot have `datad` input connections. The `datad` port must be disconnected or the operating mode must be changed to normal.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG LCELL primitive <*name*> cin0, cin1, cout0, and cout1 ports cannot be used in design entry

---

CAUSE: The `cin0`, `cin1`, `cout0`, and `cout1` ports of the specified WYSIWYG LCELL primitive are available only in the post-compilation netlists.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG LCELL primitive <*name*> fans out via carry or cascade chains to more than one destination WYSIWYG LCELL primitive

---

CAUSE: The `cout` and `cascout` ports of the specified WYSIWYG LCELL primitive are feeding different WYSIWYG LCELL primitives. The carry or cascade chain fan-out must be converted to regular output ports or the destination logic must be merged into a single WYSIWYG LCELL primitive.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG LCELL primitive <*name*> feeds more than one logic cell via carry or cascade chains

---

CAUSE: In the current design file, the `cascout` and `cout` ports of the specified WYSIWYG LCELL primitive feed different logic cells. However, the `cascout` and `cout` ports of a WYSIWYG LCELL primitive must both feed the same logic cell, or the WYSIWYG LCELL primitive must use only one of the `cascout` and `cout` ports. This error can occur when the source netlist contains an error.

ACTION: Modify the design so that the `cascout` and `cout` ports of the WYSIWYG LCELL primitive both feed the same logic cell, or modify the design so that the WYSIWYG LCELL primitive uses only one of the `cascout` and `cout` ports.

WYSIWYG LCELL primitive <*name*> has cascin port that cannot be inverted, VCC, or GND

CAUSE: The specified WYSIWYG LCELL primitive has an inverted, `VCC`, or `GND` port for the `cascin` port. The inversion must be removed or a WYSIWYG LCELL primitive must be created with `cascout` of `VCC`, `GND`, or inverted signal.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG LCELL primitive <*name*> has cin and cascin ports connected to different WYSIWYG LCELL primitives, but the ports must be connected to the same WYSIWYG LCELL primitive

---

CAUSE: The specified WYSIWYG LCELL primitive has `cin` and `cascin` ports that are connected to different WYSIWYG LCELL primitives. When the `cin` port is connected to the `cout` port, and the `cascin` port is connected to the `cascout` port, the `cout` and `cascout` ports must be in the same WYSIWYG primitive.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG LCELL primitive <*name*> has cin port that cannot be inverted, VCC, or GND

CAUSE: The specified WYSIWYG LCELL primitive has an inverted, `VCC`, or `GND` port for the `cin` port of this WYSIWYG LCELL primitive. The inversion must be removed or a WYSIWYG LCELL primitive must be created with a `cout` of `VCC`, `GND`, or inverted signal.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG LCELL primitive <*name*> has cout or cascout port that feeds another WYSIWYG LCELL primitive's input port other than cin or cascin port

---

CAUSE: The specified WYSIWYG LCELL primitive has a `cout` or `cascout` port that is connected to the input port other than `cin` or `cascin` of another WYSIWYG LCELL primitive. The port connections must be removed, or the `cin` or `cascin` port of the other WYSIWYG LCELL primitive must be connected to a `cin` or `cascin` port.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG LCELL primitive <*name*> has illegal cout port with fan-out when in normal mode

---

CAUSE: The specified WYSIWYG LCELL primitive has an illegal `cout` port with fan-out when in normal mode. The WYSIWYG primitive cannot have a `cout` port when in normal mode. The `cout` port must be removed.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG LCELL primitive <*name*> has illegal value for LUT_MASK parameter

---

CAUSE: The specified WYSIWYG LCELL primitive has a `LUT_MASK` parameter value with an illegal format. The value must be changed to a four-digit hexadecimal number.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG LCELL primitive <*name*> has illegal value for PACKED_MODE parameter

---

CAUSE: The specified WYSIWYG LCELL primitive has an illegal `PACKED_MODE` parameter value. The illegal parameter value must be replaced with a legal parameter value.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG LCELL primitive <*name*> has LUT_MASK parameter that is not set to VCC or GND, but has no data inputs

---

CAUSE: The specified WYSIWYG LCELL primitive has a `LUT_MASK` parameter that is not set to `VCC` or `GND`, but has no data inputs. The `LUT_MASK` parameter, which is not set to `VCC` or `GND`, must have data inputs.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG LCELL primitive <*name*> has PACKED_MODE parameter set to TRUE, but uses cin or cout port

---

CAUSE: The specified WYSIWYG LCELL primitive has a `PACKED_MODE` parameter that is set to `TRUE`, but uses the `cin` port or the `cout` port. The WYSIWYG primitive cannot use the `cin` port or the `cout` port if the `PACKED_MODE` parameter is set to `TRUE`. Either the `cin` and `cout` ports must not be used, or the `PACKED_MODE` parameter must be set to `FALSE`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG LCELL primitive <*name*> has PACKED_MODE parameter value set to TRUE, but does not use datac port

---

CAUSE: The specified WYSIWYG LCELL primitive has a `PACKED_MODE` parameter value that is set to `TRUE`, but does not use a `datac` port. The WYSIWYG primitive cannot have a `PACKED_MODE` parameter value that is set to `TRUE` when it does not use a `datac` port. Either the `PACKED_MODE` parameter value must be changed to `FALSE`, or a `datac` port must be used.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG LCELL primitive <*name*> has PACKED_MODE parameter value set to TRUE, but does not use datad port

---

CAUSE: The specified WYSIWYG LCELL primitive has a `PACKED_MODE` parameter value that is set to `TRUE`, but does not use a `datad` port. The WYSIWYG primitive cannot have a `PACKED_MODE` parameter value that is set to `TRUE` when it does not use a `datad` port. Either the `PACKED_MODE` parameter value must be changed to `FALSE`, or a `datad` port must be added.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG LCELL primitive <*name*> has synchronous control signals that are not part of a carry chain.

CAUSE: The specified WYSIWYG LCELL primitive has its `.sclr` and/or `.sload` ports connected, but it is not part of carry chain. This may have an affect on the ability to fit the design.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG LCELL primitive <*name*> is dependent on unconnected inputs

---

CAUSE: The specified WYSIWYG LCELL primitive has a `LUT_MASK` parameter value that is sensitive to changes on an input port that is not connected. The `LUT_MASK` parameter must be simplified, or the appropriate input ports must be connected.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG LCELL primitive <*name*> is operating in counter mode, but this mode no longer exists for the current device family -- setting the operating mode to arithmetic

---

CAUSE: The specified WYSIWYG LCELL primitive is set to counter mode, but the current device family no longer supports this mode. The Compiler has automatically changed the operating mode to arithmetic.

ACTION: No action is required.

WYSIWYG LCELL primitive <*name*> must have LUT_MASK parameter

---

CAUSE: The specified WYSIWYG LCELL primitive does not have the required `LUT_MASK` parameter. A `LUT_MASK` parameter with a legal value must be specified.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG LCELL primitive <*name*> must not use both regout and combout ports

---

CAUSE: The specified WYSIWYG LCELL primitive in a FLEX 6000 design uses both the `combout` and `regout` ports, but only one of these ports is allowed for FLEX 6000 devices. Either the `combout` port or the `regout` port must be removed, so only one of the ports is used.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG LCELL primitive <*name*> must not use combout port when in counter mode

---

CAUSE: The specified WYSIWYG LCELL primitive uses a `combout` port, but the `OPERATION_MODE` parameter is set to `COUNTER`. The WYSIWYG primitive must not use a `combout` port when it is in counter mode. Either the `combout` port must not be used, or the `OPERATION_MODE` parameter must be set to `NORMAL` or `ARITHMETIC`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG LCELL primitive <*name*> must not use datac port when in arithmetic mode

---

CAUSE: The specified WYSIWYG LCELL primitive uses a `datac` port, but the `OPERATION_MODE` parameter is set to `ARITHMETIC`. The WYSIWYG primitive must not use a `datac` port when it is in arithmetic mode. Either the `datac` port must not be used or the `OPERATION_MODE` parameter must be set to `NORMAL`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG LCELL primitive <*name*> must not use datad port when in arithmetic mode

---

CAUSE: The specified WYSIWYG LCELL primitive, has a `datad` port, but its `OPERATION_MODE` parameter is set to `ARITHMETIC`. The WYSIWYG LCELL primitive must not have a `datad` port when it is in arithmetic mode. Either the `datad` port must be removed from the WYSIWYG LCELL primitive, or the `OPERATION_MODE` parameter must be set to `NORMAL`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG LCELL primitive <*name*> must use an adata port when aload port is used

---

CAUSE: The specified WYSIWYG LCELL primitive uses an `aload` port, but does not use an `adata` port. An `adata` port must be used, or the `sload` port must not be used.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG LCELL primitive <*name*> must use sdata port when sload port is used

---

CAUSE: The specified WYSIWYG LCELL primitive uses an `sload` port, but does not use an `sdata` port. An `sdata` port must be used, or the `sload` port must not be used.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG LCELL primitives have synchronous control signals that are not part of a carry chain

---

CAUSE: One or more WYSIWYG LCELL primitives have `.sclr` and/or `.sload` ports, but are not part of a carry chain. This condition may have an affect on the ability to fit the design.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG LVDS receiver primitive <*name*> has illegal value for CHANNEL_WIDTH parameter

---

CAUSE: The specified WYSIWYG LVDS receiver primitive has an illegal value for the `CHANNEL_WIDTH` parameter. A legal value for the `CHANNEL_WIDTH` parameter must be specified.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG LVDS transmitter primitive <*name*> has illegal CHANNEL_WIDTH parameter

---

CAUSE: The specified WYSIWYG LVDS transmitter primitive has an illegal value for the `CHANNEL_WIDTH` parameter. A legal value for the `CHANNEL_WIDTH` parameter must be specified.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG LVDS transmitter primitive <*name*> has value for CHANNEL_WIDTH parameter that does not correspond to the number of datain port connections

---

CAUSE: The specified WYSIWYG LVDS transmitter primitive has a value for its `CHANNEL_WIDTH` parameter that does not correspond to the number of `datain` port connections. Either some `datain` port connections must be added or removed or the value for the WYSIWYG LVDS transmitter primitive's `CHANNEL_WIDTH` parameter must be changed.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG MCELL primitive <*name*> cannot use both aclr and paclr ports

---

CAUSE: The specified WYSIWYG MCELL primitive uses both the `aclr` and `paclr` ports; however, it can use only one of the two ports.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG MCELL primitive <*name*> cannot use both clk and pclk ports

---

CAUSE: The specified WYSIWYG MCELL primitive uses both the `clk` and `pclk` ports; however, it can use only one of the two ports.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG MCELL primitive <*name*> has illegal combination of mode parameter settings -- cannot be in VCC mode and combinatorial mode when parallel expander mode is turned off

---

CAUSE: The specified WYSIWYG MCELL primitive is set to VCC mode (that is, the `OPERATION_MODE` parameter is set to `VCC`) and is set to combinatorial mode (that is, the `OUTPUT_MODE` parameter is set to `COMB`), but has parallel expander mode turned off (that is, the `PEXP_MODE` parameter is set to `OFF`). When the WYSIWYG MCELL primitive is in VCC mode and is not in parallel expander mode, it can have no data imputs and can implement only a TFF or a fast input register, and therefore cannot also be in combinatorial mode. At least one of the parameter settings must be changed.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG MCELL primitive <*name*> has illegal combination of mode parameters -- cannot be in XOR or XNOR mode, TFF mode, and parallel expander mode

---

CAUSE: The specified WYSIWYG MCELL primitive is in either `XOR` or `XNOR` mode (that is, the `OPERATION_MODE` parameter is set to `XOR` or `XNOR`) and is also in `TFF` mode (that is, the `REGISTER_MODE` parameter is set to `TFF`) and parallel expander mode (that is, the `PEXP_MODE` parameter is set to `ON`). The `TFF` mode can be used only when the `OPERATION_MODE` parameter is set to `NORMAL`, `INVERT`, or `VCC`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG MCELL primitive <*name*> has illegal value for PEXP_MODE parameter -- value must be ON or OFF

CAUSE: The specified WYSIWYG MCELL primitive has an illegal `PEXP_MODE` parameter value. The `PEXP_MODE` parameter must be set to `ON` or `OFF`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG MCELL primitive <*name*> has illegal value for POWER_UP parameter -- value must be HIGH or LOW

CAUSE: The specified WYSIWYG MCELL primitive has an illegal `POWER_UP` parameter value. The `POWER_UP` parameter must be `HIGH` or `LOW`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG MCELL primitive <*name*> has illegal value for REGISTER_MODE parameter -- value must be DFF or TFF

CAUSE: The specified WYSIWYG MCELL primitive has an illegal `REGISTER_MODE` parameter value. The `REGISTER_MODE` parameter must be set to `DFF` or `TFF`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG MCELL primitive <*name*> register must use fpin fast input port if it is in register mode, DFF mode, and VCC mode

---

CAUSE: The specified WYSIWYG MCELL primitive is in register mode (that is, the `OUTPUT_MODE` parameter is set to `REG`), DFF mode (that is, the `REGISTER_MODE` parameter is set to `DFF`) and VCC mode (that is, the `OPERATION_MODE` parameter is set to `VCC`), but does not use the `fpin` fast input port. The `fpin` port must be used when the WYSIWYG primitive is in register mode, DFF mode, and VCC mode.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> belongs to wrong device family

CAUSE: You created a WYSIWYG primitive with a different family type the device family that you specified.

ACTION: Specify the appropriate device family and recompile the design.

WYSIWYG primitive <*name*> cannot have <*name*> parameter set to <*value*> <*text*>

CAUSE: The specified WYSIWYG primitive has the specified parameter, which is set to the specified value, but the parameter cannot be set to that value for the specified reason.

ACTION: Modify the design to remove the parameter value for this WYSIWYG primitive or specify a legal parameter value. If the value is correct, and you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> cannot have BIT_NUMBER parameter value greater than or equal to LOGICAL_RAM_WIDTH value

---

CAUSE: The specified WYSIWYG primitive has either an incorrect `BIT_NUMBER` parameter value or an incorrect `LOGICAL_RAM_WIDTH` value such that the `BIT_NUMBER` parameter value is greater than or equal to the `LOGICAL_RAM_WIDTH` value. The `BIT_NUMBER` parameter value, which is zero-based, or `LOGICAL_RAM_WIDTH` value must be modified so that the `BIT_NUMBER` parameter value is less than the `LOGICAL_RAM_WIDTH` value.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> cannot have clk port when regout port is not used

---

CAUSE: The specified WYSIWYG primitive uses a `clk` port, but does not use a `regout` port. The WYSIWYG primitive cannot use a `clk` port if the `regout` port is not used. Either a `regout` port must also be used or the `clk` port must not be used.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> cannot have datain port when in read-only mode

---

CAUSE: The specified WYSIWYG primitive has a `datain` port when in read-only mode. The WYSIWYG primitive cannot have a `datain` port when in read-only mode. The `datain` port must be removed from the WYSIWYG primitive.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> cannot have FIRST_PATTERN_BIT parameter value greater than or equal to LOGICAL_CAM_WIDTH parameter value

---

CAUSE: The specified WYSIWYG primitive contains either an incorrect `FIRST_PATTERN_BIT` or an incorrect `LOGICAL_CAM_WIDTH` parameter value such that the `FIRST_PATTERN_BIT` value is greater than or equal to the `LOGICAL_CAM_WIDTH` value. The `FIRST_PATTERN_BIT` value, which is zero-based, or the `LOGICAL_CAM_WIDTH` value must be modified so that the `FIRST_PATTERN_BIT` parameter value is less than the `LOGICAL_CAM_WIDTH` parameter value.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> cannot have pexpout port when its OPERATION_MODE parameter is not set to PTERM_EXP or PACKED_PTERM_EXP

CAUSE: The specified WYSIWYG primitive has a `pexpout` port even though its `OPERATION_MODE` parameter is not set to `PTERM_EXP` or `PACKED_PTERM_EXP`. The WYSIWYG primitive cannot have a `pexpout` port when the `OPERATION_MODE` parameter is not set to `PTERM_EXP` or `PACKED_PTERM_EXP`. The `pexpout` port must be removed.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> cannot have re port when in read-only mode

---

CAUSE: The specified WYSIWYG primitive, which is in read-only mode, has a `re` port. The WYSIWYG primitive cannot have a `re` port when in read-only mode. The `re` port must be removed from the WYSIWYG primitive.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> cannot have re port when in single-port mode

---

CAUSE: The specified WYSIWYG primitive, which is in single-port mode, has a `re` port. The WYSIWYG primitive cannot have a `re` port when in single-port mode. The `re` port must be removed from the WYSIWYG primitive.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> cannot have READ_ADDRESS_CLEAR parameter when in deep RAM mode

---

CAUSE: The specified WYSIWYG primitive has a `READ_ADDRESS_CLEAR` parameter when in deep RAM mode. The WYSIWYG primitive cannot have a `READ_ADDRESS_CLEAR` parameter when in deep RAM mode. Either the deep RAM mode must be turned off or the `READ_ADDRESS_CLEAR` parameter should be set to `NONE`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> cannot have waddr port when in read-only mode

---

CAUSE: The specified WYSIWYG primitive, which is in read-only mode, has a `waddr` port. The WYSIWYG primitive cannot have a `waddr` port when in read-only mode. The `waddr` port must be removed from the WYSIWYG primitive.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> cannot have we port when in read-only mode

---

CAUSE: The specified WYSIWYG primitive, which is in read-only mode, has a `we` port. The `we` port must be removed from the WYSIWYG primitive.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> cannot use clk port when its OPERATION_MODE parameter is set to PTERM_EXP

CAUSE: The specified WYSIWYG primitive uses a `clk` port when its `OPERATION_MODE` is set to `PTERM_EXP`. The WYSIWYG primitive cannot use a `clk` port when the `OPERATION_MODE` parameter is set to `PTERM_EXP`. The `clk` port must not be used or the `OPERATION_MODE` parameter must be set to something other than `PTERM_EXP`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> cannot use clk port when its OUTPUT_MODE parameter is set to COMB

CAUSE: The specified WYSIWYG primitive uses a `clk` port even though its `OUTPUT_MODE` parameter is set to `COMB`. The WYSIWYG primitive cannot use a `clk` port when its `OUTPUT_MODE` parameter is set to `COMB`. Either the `clk` port must not be used or the `OUTPUT_MODE` parameter must be set to `REG`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> cannot use dataout port when OPERATION_MODE parameter is set to PTERM_EXP

CAUSE: The specified WYSIWYG primitive uses a `dataout` port, but the `OPERATION_MODE` parameter is set to `PTERM_EXP`. The WYSIWYG primitive cannot use a `dataout` port when the `OPERATION_MODE` parameter is set to `PTERM_EXP`. The `dataout` port must not be used, or the `OPERATION_MODE` parameter must be set to a value other than `PTERM_EXP`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> cannot use pterm5 port when in XOR mode and when pexpout port is not used

---

CAUSE: The specified WYSIWYG primitive is in XOR mode (that is, the `OPERATION_MODE` parameter is set to `XOR`), and does not use the `pexpout` port, but uses the `pterm5` port. The `pterm5` port must not be used if the `pexpout` port is not used and the WYSIWYG primitive is in XOR mode.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> cannot use registered dataout port when in deep RAM mode

---

CAUSE: The specified WYSIWYG primitive, which is in deep RAM mode, is using a registered `dataout` port. In this mode, the `dataout` port cannot be registered. The `dataout` port must be changed to non-registered or the WYSIWYG primitive's `DATA_OUT_CLEAR` parameter must be set to `NONE`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> clock port must be fed by input pin, and must not be fed by any logic gates

---

CAUSE: The specified WYSIWYG primitive uses a `clock` port that is fed by a logic gate. The `clock` port must be fed by an input pin and must not be fed by any logic gates.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> does not have OUTPUT_MODE parameter value specified and is defaulting to COMB

CAUSE: The specified WYSIWYG primitive does not have the required `OUTPUT_MODE` parameter value. The WYSIWYG primitive defaulted to `COMB` because a `clk` port was not specified.

ACTION: No action is required. However, if you are encountering problems and are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> does not have OUTPUT_MODE parameter value specified and is defaulting to REG

CAUSE: The specified WYSIWYG primitive does not have the required `OUTPUT_MODE` parameter value. The WYSIWYG primitive defaulted to `REG` because a `clk` port was specified.

ACTION: No action is required. However, if you are encountering problems and you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has <*name*> port that must be connected <*text*>

CAUSE: The specified WYSIWYG primitive has the specified port, which is not connected, but the port must be connected for the specified reason.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has <*name*> port that must not be connected <*text*>

CAUSE: The specified WYSIWYG primitive has the specified port, which is connected, but the port must not be connected for the specified reason.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has <*name*> port that should be connected <*text*>

CAUSE: The specified WYSIWYG primitive has the specified port, which is not connected, but the port should be connected for the specified reason.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has <*name*> port with a specified width of <*number*>, but has <*number*> more bits connected than the specified width

---

CAUSE: The specified WYSIWYG primitive has the specified port with the specified width, but the number of connected bits exceeds the number of bits specified in the width. Either the extra bits of the port must not be connected, or the width specification for the port must be larger to accommodate the extra bits.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has <*name*> port with specified width of <*number*>, but only <*number*> bits are connected

---

CAUSE: The specified WYSIWYG primitive has the specified port with the specified width, but the number of bits that are connected is less than the specified width. Either all of the bits of the port must be connected, or the width specification for the port must be smaller.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has aclr port, but does not have regout port

---

CAUSE: The specified WYSIWYG primitive has an `aclr` port, but does not have a `regout` port. The WYSIWYG primitive cannot have an `aclr` port without having a `regout` port as well. A `regout` port must be added to or the `aclr` port must be removed from the WYSIWYG primitive.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has aclr port, but does not use its register

---

CAUSE: The specified WYSIWYG primitive uses an `aclr` port, but does not use its register. Either a `clk` port must also be used or the `aclr` port must not be used.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has aload port, but does not have regout port

---

CAUSE: The specified WYSIWYG primitive has an `aload` port, but does not have a `regout` port. The WYSIWYG primitive cannot have an `aload` port without having a `regout` port as well. A `regout` port must be added to or the `aload` port must be removed from the WYSIWYG primitive.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has aload port, but does not use its register

---

CAUSE: The specified WYSIWYG primitive has an `aload` port, but does not use its register. Either a `clk` port must also be used or the `aload` port must not be used.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has apre port, but does not have regout port

---

CAUSE: The specified WYSIWYG primitive has an `apre` port, but does not have a `regout` port. The WYSIWYG primitive cannot have an `apre` port without having a `regout` port as well. A `regout` port must be added to or the `apre` port must be removed from the WYSIWYG primitive.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has apre port, but does not use its register

---

CAUSE: The specified WYSIWYG primitive uses an `apre` port, but does not use its register. Either a `clk` port must also be used or the `apre` port must not be used.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has clk port, but does not contain regout port

---

CAUSE: The specified WYSIWYG primitive has a `clk` port, but does not have a `regout` port. The WYSIWYG primitive cannot have a `clk` port without having a `regout` port as well. A `regout` port must be added to or the `clk` port must be removed from the WYSIWYG primitive.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has CLOCK_ENABLE_MODE parameter value that is set to TRUE, but does not use its dataa port

---

CAUSE: The specified WYSIWYG primitive has a `CLOCK_ENABLE_MODE` parameter value that is set to `TRUE`, but does not use its `dataa` port. The WYSIWYG primitive must use its `dataa` if its `CLOCK_ENABLE_MODE` parameter value is set to `TRUE`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has ena port, but does not have regout port

---

CAUSE: The specified WYSIWYG primitive has an `ena` port, but does not have a `regout` port. The WYSIWYG primitive cannot have an `ena` port without having a `regout` port as well. A `regout` port must be added to or the `ena` port must be removed from the WYSIWYG primitive.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has ena port, but does not use its register

---

CAUSE: The specified WYSIWYG primitive uses an `ena` port, but does not use its register. Either a `clk` port must also be used or the `ena` port must not be used.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has ena0 port, but does not have clk0 port

---

CAUSE: The specified WYSIWYG primitive has an `ena0` port, but does not have a `clk0` port. Either a `clk0` port must be added to or the `ena0` port must be removed from the WYSIWYG primitive.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has ena1 port, but does not have clk1 port

---

CAUSE: The specified WYSIWYG primitive has an `ena` port, but does not have a `clk1` port. Either a `clk1` port must be added to or the `ena1` port must be removed from the WYSIWYG primitive.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has illegal pexpin port -- cannot be VCC, GND, inverted, direct from pin, or from non-parallel expander output port on another WYSIWYG primitive

---

CAUSE: The specified WYSIWYG primitive has an illegal `pexpin` port that is either `VCC`, `GND`, inverted, direct from pin, or from a non-`pexpout` port on another WYSIWYG primitive. The illegal signal must be replaced with a legal signal.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has illegal value for <*name*> parameter

---

CAUSE: You specified an illegal value for the parameter of the specified WYSIWYG primitive.

ACTION: Modify the design to correct the parameter value for this WYSIWYG primitive. If the value is correct, and you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has illegal value for <*name*> parameter -- value must be a number

---

CAUSE: The specified WYSIWYG primitive has the specified parameter, but the parameter has a value that is not a number. The value of the specified parameter must be a number.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has illegal value for DATA_IN_CLEAR parameter

---

CAUSE: The specified WYSIWYG primitive has an illegal `DATA_IN_CLEAR` parameter value. The `DATA_IN_CLEAR` parameter value must be changed to either `CLEAR0` or `NONE`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has illegal value for DATA_IN_CLOCK parameter

---

CAUSE: The specified WYSIWYG primitive has an illegal `DATA_IN_CLOCK` parameter value. The `DATA_IN_CLOCK` parameter value must be changed to either `CLOCK0` or `NONE`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has illegal value for DATA_OUT_CLEAR parameter

---

CAUSE: The specified WYSIWYG primitive has an illegal `DATA_OUT_CLEAR` parameter value. The `DATA_OUT_CLEAR` parameter value must be changed to `CLEAR0`, `CLEAR1`, or `NONE`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has illegal value for DATA_OUT_CLOCK parameter

---

CAUSE: The specified WYSIWYG primitive has an illegal `DATA_OUT_CLOCK` parameter value. The `DATA_OUT_CLOCK` parameter value must be changed to `CLOCK0`, `CLOCK1`, or `NONE`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has illegal value for INVERT_PTERM1_MODE parameter

---

CAUSE: The specified WYSIWYG primitive has an illegal `INVERT_PTERM1_MODE` parameter value. The illegal parameter value must be replaced with a legal parameter value.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has illegal value for OPERATION_MODE parameter

---

CAUSE: The specified WYSIWYG primitive has an illegal `OPERATION_MODE` parameter value. The illegal parameter value must be replaced with a legal parameter value.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has illegal value for OUTPUT_CLEAR parameter

---

CAUSE: The specified WYSIWYG primitive has an illegal `OUTPUT_CLEAR` parameter value. The `OUTPUT_CLEAR` parameter value must be changed to `CLEAR0`, `CLEAR1`, or `NONE`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has illegal value for OUTPUT_CLOCK parameter

---

CAUSE: The specified WYSIWYG primitive has an illegal `OUTPUT_CLOCK` parameter value. The `OUTPUT_CLOCK` parameter value must be changed to `CLOCK0`, `CLOCK1`, or `NONE`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has illegal value for OUTPUT_MODE parameter

---

CAUSE: The specified WYSIWYG primitive has an illegal `OUTPUT_MODE` parameter value. The illegal parameter value must be replaced with a legal parameter value.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has illegal value for PATTERN_WIDTH parameter

---

CAUSE: The specified WYSIWYG primitive has an illegal `PATTERN_WIDTH` parameter value. The illegal `PATTERN_WIDTH` parameter value must be replaced with a legal parameter value.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has illegal value for READ_ADDRESS_CLEAR parameter

---

CAUSE: The specified WYSIWYG primitive has an illegal `READ_ADDRESS_CLEAR` parameter value. The `READ_ADDRESS_CLEAR` parameter value must be changed to `CLEAR0`, `CLEAR1`, or `NONE`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has illegal value for READ_ADDRESS_CLOCK parameter

---

CAUSE: The specified WYSIWYG primitive has an illegal `READ_ADDRESS_CLOCK` parameter value. The `READ_ADDRESS_CLOCK` parameter value must be changed to `CLOCK0`, `CLOCK1`, or `NONE`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has illegal value for READ_ENABLE_CLEAR parameter

---

CAUSE: The specified WYSIWYG primitive has an illegal `READ_ENABLE_CLEAR` parameter value. The `READ_ENABLE_CLEAR` parameter value must be changed to `CLEAR0`, `CLEAR1`, or `NONE`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has illegal value for READ_ENABLE_CLOCK parameter

---

CAUSE: The specified WYSIWYG primitive has an illegal `READ_ENABLE_CLOCK` parameter value. The `READ_ENABLE_CLOCK` parameter value must be changed to `CLOCK0`, `CLOCK1`, or `NONE`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has illegal value for WRITE_ADDRESS_CLEAR parameter

---

CAUSE: The specified WYSIWYG primitive has an illegal `WRITE_ADDRESS_CLEAR` parameter value. The `WRITE_ADDRESS_CLEAR` parameter value must be changed to either `CLEAR0` or `NONE`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has illegal value for WRITE_ENABLE_CLEAR parameter

---

CAUSE: The specified WYSIWYG primitive has an illegal `WRITE_ENABLE_CLEAR` parameter value. The `WRITE_ENABLE_CLEAR` parameter value must be changed to either `CLEAR0` or `NONE`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has illegal value for WRITE_LOGIC_CLEAR parameter

---

CAUSE: The specified WYSIWYG primitive has an illegal `WRITE_LOGIC_CLEAR` parameter value. The `WRITE_LOGIC_CLEAR` parameter value must be changed to either `CLEAR0` or `NONE`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has illegal value for WRITE_LOGIC_CLOCK parameter

---

CAUSE: The specified WYSIWYG primitive has an illegal `WRITE_LOGIC_CLOCK` parameter value. The `WRITE_LOGIC_CLOCK` parameter value must be changed to either `CLOCK0` or `NONE`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has incompatible parameters

CAUSE: Some parameters of the specified WYSIWYG primitive do not match, do not make sense, or do not fall within the bounds of other related parameters. All the parameters in the WYSIWYG primitive must have values that agree with each other and must be changed so that they match.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has inconsistent parameter values <*name*> and <*name*>

CAUSE: The specified WYSIWYG primitive has the specified parameter values, which are inconsistent with each other. The parameters must have consistent values.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has incorrect Hexadecimal (Intel-Format) File name <*name*> for LPM_FILEX or INIT_FILEX parameter

---

CAUSE: The specified WYSIWYG primitive has the specified Hexadecimal (Intel-Format) File (**.hex**) name that does not follow the format required for the `LPM_FILEX` or `INIT_FILEX` parameter. If the `LPM_FILEX` or `INIT_FILEX` parameter is used, the file name must be of the form ***<LPM_FILEX>_xu.hex*** or ***<INIT_FILE>_xu.hex***, respectively.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has sclr port, but does not use its register

---

CAUSE: The specified WYSIWYG primitive uses an `sclr` port, but does not use its register. Either a `clk` port must also be used or the `sclr` port must not be used.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> has sload port, but does not use its register

---

CAUSE: The specified WYSIWYG primitive uses an `sload` port, but does not use its register. Either a `clk` port must also be used or the `sload` port must not be used.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> is in unsupported dual-port mode -- must have waddr and wena ports that are both either registered or non-registered

---

CAUSE: The specified WYSIWYG primitive, which is in dual-port mode, has either a registered `waddr` port and a non-registered `wena` port, or a non-registered `waddr` port and a registered `wena` port. The MAX+PLUS II Advanced Synthesis software does not support this combination of registered and non-registered ports in dual-port mode. The WYSIWYG primitive must have both the `waddr` and `wena` ports either registered or non-registered.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> is in unsupported ROM mode -- must have ports that are both either registered or non-registered

---

CAUSE: The specified WYSIWYG primitive, which is in ROM mode, has ports that are not all registered or non-registered. The MAX+PLUS II Advanced Synthesis software does not support a combination of registered and non-registered ports in ROM mode. The WYSIWYG primitive must have all its ports either registered or non-registered.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> is in unsupported single-port mode -- must have waddr and raddr ports that are both either registered or non-registered

---

CAUSE: The specified WYSIWYG primitive, which is in single-port mode, has either a registered `waddr` port and a non-registered `raddr` port, or a non-registered `waddr` port and a registered `raddr` port. The MAX+PLUS II Advanced Synthesis software does not support this combination of registered and non-registered ports in single-port mode. The WYSIWYG primitive must have both the `waddr` and `raddr` ports either registered or non-registered.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> is missing required <*name*> parameter

---

CAUSE: The specified WYSIWYG primitive does not use the specified parameter, but the parameter is required.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> is missing value for <*name*> parameter. Legal values for <*name*> parameter are <*values*>.

CAUSE: The specified WYSIWYG primitive has the specified parameter, which is missing a value. The parameter must have a value because it is a required parameter. The message text specifies possible legal values.

ACTION: Modify the design to correct the parameter value for the WYSIWYG primitive. If the value is correct, and you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must have <*name*> parameter set to <*value*> <*text*>

CAUSE: The specified WYSIWYG primitive must have the specified parameter set to the specified value for the specified reason.

ACTION: Modify the design to make sure the WYSIWYG primitive has the specified parameter set to the specified value. If the value is correct, and you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must have BIT_NUMBER parameter

CAUSE: The specified WYSIWYG primitive does not have a required `BIT_NUMBER` parameter. A `BIT_NUMBER` parameter with a legal value must be specified.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must have clk port

CAUSE: The specified WYSIWYG primitive uses a register that requires a `clk` port. Either a `clk` port must be added to the WYSIWYG primitive or the `regout` port must be changed to a `combout` port.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must have clk port when OPERATION_MODE parameter is set to TFF or TBARFF

CAUSE: The specified WYSIWYG primitive does not have a `clk` port even though its `OPERATION_MODE` parameter is set to `TFF` or `TBARFF`. The WYSIWYG primitive must have a `clk` port when the `OPERATION_MODE` parameter is set to `TFF` or `TBARFF`. A `clk` port must be specified.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must have clk port when OUTPUT_MODE parameter is set to REG

CAUSE: The specified WYSIWYG primitive does not have a `clk` port even though its `OUTPUT_MODE` parameter is set to `REG`. The WYSIWYG primitive must have a `clk` port when the `OUTPUT_MODE` parameter is set to `REG`. Either a `clk` port must be added or the `OUTPUT_MODE` parameter must be set to `COMB`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must have clk0 port

CAUSE: Although the specified WYSIWYG primitive has a `CLOCK0` parameter, the `clk0` port does not exist. Either the `CLOCK0` parameter must be removed or the `clk0` port must be added to the WYSIWYG primitive.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must have clk1 port

CAUSE: Although the specified WYSIWYG primitive has a `CLOCK1` parameter, the `clk1` port does not exist. Either the `CLOCK1` parameter must be removed or a `clk1` port must be added to the WYSIWYG primitive.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must have CLOCK_ENABLE_MODE parameter

---

CAUSE: The specified WYSIWYG primitive does not have the required `CLOCK_ENABLE_MODE` parameter. A `CLOCK_ENABLE_MODE` parameter with a legal value must be specified.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must have clr0 port

CAUSE: Although the specified WYSIWYG primitive has a `CLEAR0` parameter, the `clr0` port does not exist. Either `CLEAR0` parameter must be removed or a `clr0` port must be added to the WYSIWYG primitive.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must have clr1 port

CAUSE: Although the specified WYSIWYG primitive has a `CLEAR1` parameter, the `clr1` port does not exist. Either `CLEAR1` parameter must be removed or a `clr1` port must be added to the WYSIWYG primitive.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must have DATA_OUT_CLOCK parameter when value is specified for its DATA_OUT_CLEAR parameter

---

CAUSE: The specified WYSIWYG primitive, which has a value specified for its `DATA_OUT_CLEAR` parameter, does not have a `DATA_OUT_CLOCK` parameter. A `DATA_OUT_CLOCK` parameter with a legal value must be specified for the WYSIWYG primitive when a value has been specified for its `DATA_OUT_CLEAR` parameter.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must have FIRST_ADDRESS parameter

CAUSE: The specified WYSIWYG primitive does not have a required `FIRST_ADDRESS` parameter. A `FIRST_ADDRESS` parameter with a legal value must be specified.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must have FIRST_PATTERN_BIT parameter

---

CAUSE: The specified WYSIWYG primitive does not have a required `FIRST_PATTERN_BIT` parameter. A `FIRST_PATTERN_BIT` parameter with a legal value must be specified.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must have LAST_ADDRESS parameter

CAUSE: The specified WYSIWYG primitive does not have a required LAST_ADDRESS parameter. A LAST_ADDRESS parameter with a legal value must be specified.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must have legal value for ADDRESS_WIDTH parameter

---

CAUSE: The specified WYSIWYG primitive has an illegal `ADDRESS_WIDTH` parameter value. An `ADDRESS_WIDTH` parameter with a legal value must be specified.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must have LOGICAL_CAM_NAME parameter

---

CAUSE: The specified WYSIWYG primitive does not have a `LOGICAL_CAM_NAME` parameter. A `LOGICAL_CAM_NAME` parameter with a legal value must be specified.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must have LOGICAL_RAM_WIDTH parameter

---

CAUSE: The specified WYSIWYG primitive does not have a required `LOGICAL_RAM_WIDTH` parameter value. A `LOGICAL_RAM_WIDTH` parameter with a legal value must be specified.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must have OPERATION_MODE parameter

---

CAUSE: The specified WYSIWYG primitive does not have a required `OPERATION_MODE` parameter. An `OPERATION_MODE` parameter with a legal value must be specified.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must have PATTERN_WIDTH parameter

CAUSE: The specified WYSIWYG primitive does not have a required `PATTERN_WIDTH` parameter. A `PATTERN_WIDTH` parameter with a legal value must be specified.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must have pterm0 port when in packed product term expander mode

---

CAUSE: The specified WYSIWYG primitive does not have a `pterm0` port, even though it is in packed product term expander mode. The WYSIWYG primitive must have a `pterm0` port when it is in packed product term expander mode.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must have READ_ADDRESS_CLEAR parameter when value is specified for its READ_ADDRESS_CLOCK parameter

---

CAUSE: The specified WYSIWYG primitive, which has a value specified for its `READ_ADDRESS_CLOCK` parameter, does not have a `READ_ADDRESS_CLEAR` parameter. A `READ_ADDRESS_CLEAR` parameter with a legal value must be specified for the WYSIWYG primitive when a value has been specified for its `READ_ADDRESS_CLOCK` parameter.

ACTION: If you are using EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must have SUM_LUTC_INPUT parameter set to CIN if cin port is used when in normal mode

---

CAUSE: The specified WYSIWYG primitive has a `cin` port, and has the `OPERATION_MODE` parameter set to `NORMAL`, but the `SUM_LUTC_INPUT` parameter is not set to `CIN`. The `SUM_LUTC_INPUT` parameter must be set to `CIN`, the `cin` port must not be used, or the `OPERATION_MODE` parameter must be set to a value other than `NORMAL`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must not use pxor port in non-XOR mode

---

CAUSE: The specified WYSIWYG primitive uses a `pxor` port, but is not in XOR mode (that is, the `OPERATION_MODE` parameter is set to something other than `XOR`). The `pxor` port is not supported for non-XOR modes. Either the `pxor` port must not be used, or the `OPERATION_MODE` parameter must be set to `XOR`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must not use regcascin port when not in register cascade mode

---

CAUSE: The specified WYSIWYG primitive is not in register cascade mode (that is, the `REGISTER_CASCADE_MODE` parameter is set to `OFF`), but uses the `regcascin` port. The regcascin port must only be used in register cascade mode. Either the `regcascin` port must not be used, or the `REGISTER_CASCADE_MODE` parameter must be set to `ON`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must use all pterm (pterm0 - pterm4) ports when in XOR mode and when pexpout port is not used

---

CAUSE: The specified WYSIWYG primitive is in XOR mode (that is, the `OPERATION_MODE` parameter is set to `XOR`), and does not use the `pexpout` port, but is not using any of the `pterm`*n* (`pterm0` - `pterm4`) ports. All of the `pterm`*n* ports must be used if the `pexpout` port is not used and the WYSIWYG primitive is in XOR mode.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must use cin port if SUM_LUTC_INPUT is set to CIN

CAUSE: The specified WYSIWYG primitive has the `SUM_LUTC_INPUT` parameter set to `CIN`, but does not use the `cin` port. The `cin` port must be used if the `SUM_LUTC_INPUT` parameter is set to `CIN`. Either the `cin` port must be used, or the `SUM_LUTC_INPUT` parameter must be set to a value other than `CIN`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must use clk port if SUM_LUTC_INPUT parameter is set to QFBK

CAUSE: The specified WYSIWYG primitive has the `SUM_LUTC_INPUT` parameter set to `QFBK`, but does not use the `clk` port. The `clk` port must be used if the `SUM_LUTC_INPUT` parameter is set to `QFBK`. Either the `clk` port must be used, or the `SUM_LUTC_INPUT` parameter must be set to a value other than `QFBK`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must use clk0 port

CAUSE: The specified WYSIWYG primitive does not use the `clk0` port, but the `clk0` port is required.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must use clk1 port if ena1 port is used

---

CAUSE: The specified WYSIWYG primitive uses the `ena1` port, but does not use the `clk1` port. The `clk1` port must be used if the `ena1` port is used.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must use either clk or pclk port for registered output

---

CAUSE: The specified WYSIWYG primitive does not use either a `clk` or a `pclk` port, but must use one of these ports for registered output.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must use pexpin port or at least one pterm port if pexpout port is used

---

CAUSE: The specified WYSIWYG primitive uses the `pexpout` port, but does not use the `pexpin` port, and is not fed by any `pterm` (`pterm0`–`pterm4`) port. Either the pexpin port or at least one `pterm` port must be used to feed the `pexpout` port.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must use pterm5 port when in XOR mode and in DFF mode, and when using the pexpout port

---

CAUSE: The specified WYSIWYG primitive is in XOR mode and DFF mode (that is, the `OPERATION_MODE` parameter is set to `XOR` and the `REGISTER_MODE` parameter is set to `DFF`), and uses the `pexpout` port, but does not use the `pterm5` port. The `pterm5` port must be used if the `pexpout` port is used and the WYSIWYG primitive is in XOR and DFF mode.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must use pxor port when in XOR mode

---

CAUSE: The specified WYSIWYG primitive is in XOR mode (that is, the `OPERATION_MODE` parameter is set to `XOR`), but does not use the `pxor` port. Either the `pxor` port must be used, or the `OPERATION_MODE` parameter must be set to `NORMAL`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> must use regcascin port when in register cascade mode

---

CAUSE: The specified WYSIWYG primitive is in register cascade mode (that is, the `REGISTER_CASCADE_MODE` parameter is set to `ON`), but does not use the `regcascin` port. The `regcascin` port must be used in register cascade mode. Either the `regcascin` port must be used, or the `REGISTER_CASCADE_MODE` parameter must be set to `OFF`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> padio port connected with ATOM <*name*> must connect to only a single top-level pin of the same type, but feeds other logic

---

CAUSE: The `padio` port of the specified WYSIWYG primitive is connected to the specified ATOM, and feeds other logic. However, the WYSIWYG primitive must connect to only a single non-WYSIWYG top-level pin of the same type, without any logic fan-out between them. This problem may be caused by instantiating a presynthesized design from another EDA tool as part of a larger design.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> padio port is connected illegally

CAUSE: The `padio` port of the specified WYSIWYG primitive is illegally connected. WYSIWYG pins must be connected to a non-WYSIWYG top-level pin of the same type, without any logic feeding in or out between them. This problem may be caused by instantiating a presynthesized design from another EDA tool as part of a larger design.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> parameter value is out of range

CAUSE: The specified value of the WYSIWYG primitive is illegal and must be replaced with a legal value.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> sclr or sload port must be used when in synchronous mode

---

CAUSE: The specified WYSIWYG primitive is in synchronous mode (that is, the `SYNCH_MODE` parameter is set to `ON`), but does not use the `sclr` port or the `sload` port. The `sclr` port or `sload` port must be used if the `SYNCH_MODE` parameter is set to `ON`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> sum LUT mask must depend on datac input signal value if SUM_LUTC_INPUT parameter is set to QFBK

CAUSE: The specified WYSIWYG primitive has the `SUM_LUTC_INPUT` parameter set to `QFBK`, but the `LUT_MASK` parameter defines a sum LUT mask that does not depend on the datac input signal value. The sum LUT mask must depend on the datac input signal value if the `SUM_LUTC_INPUT` parameter is set to `QFBK`. Either the `LUT_MASK` parameter must be set to a different value, or the `SUM_LUTC_INPUT` parameter must be set to a value other than `QFBK`.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> uses fpin fast input port that does not feed register port -- it must feed register port. In addition, WYSIWYG primitive must not use pterm5 and pxor ports, and, if PEXP_MODE parameter is set to OFF, it must not use the pterm0 - pterm4 ports.

CAUSE: The specified WYSIWYG primitive uses a `fpin` fast input port, but the `fpin` port is not connected to the register port of another WYSIWYG primitive. The WYSIWYG primitive may also be using a `pterm5` or `pxor` port, which are not supported, or may be in non-parallel expander mode (that is, it may have the `PEXP_MODE` parameter set to `OFF`) but is using `pterm` (`pterm0–pterm4`) ports. The `fpin` input port must feed the register port of another WYSIWYG primitive. In addition, the `pterm` or `pxor` ports must not be used, and either the `PEXP_MODE` parameter must be set to `ON`, or the `pterm` (`pterm0–pterm4`) ports must not be used.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> uses pexpin port, but must use all five pterm (pterm0 - pterm4) ports

---

CAUSE: The specified WYSIWYG primitive uses the `pexpin` port, but does not use all of the `pterm`*n* (`pterm0`–`pterm4`) ports. If the `pexpin` port is used, all `pterm`*n* ports must be used.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> uses pexpout port that must feed pexpin port

---

CAUSE: The specified WYSIWYG primitive uses a `pexpout` port that is not connected to the `pexpin` port of another WYSIWYG primitive. The `pexpout` port must feed the `pexpin` port of another WYSIWYG primitive.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> uses portabyteenamasks port, but PORT_A_DATA_WIDTH parameter is not a byte multiple of width of portabyteenamasks port, and portadatain port is not used

---

CAUSE: The specified WYSIWYG primitive uses the `portabyteenamasks` port, but the `PORT_A_DATA_WIDTH` parameter value is not a byte multiple of the `portabyteenamasks` port width, and the `portadatain` port is not used. The `PORT_A_DATA_WIDTH` parameter must be a byte multiple of the `portabyteenamasks` port width and the `portadatain` port must be used; otherwise, the `portabetenamasks` port must not be used.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> uses portbbyteenamasks port, but PORT_B_DATA_WIDTH parameter is not a byte multiple of width of portbbyteenamasks port, and portbdatain port is not used

---

CAUSE: The specified WYSIWYG primitive uses the `portbbyteenamasks` port, but the `PORT_B_DATA_WIDTH` parameter value is not a byte multiple of the `portbbyteenamasks` port width and the `portbdatain` port is not used. The `PORT_B_DATA_WIDTH` parameter must be a byte multiple of the `portabyteenamasks` port width and the `portbdatain` port must be used; otherwise, the `portbbyteenamasks` port must not be used.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*name*> uses pterm5 port, but pterm5 port is not supported for non-parallel expander or VCC modes

---

CAUSE: The specified WYSIWYG primitive uses the `pterm5` port, but is also either in non-parallel expander mode (that is, the `PEXP_MODE` parameter is set to `OFF`) or in VCC mode (that is, the `OPERATION_MODE` parameter is set to `VCC`). The `pterm5` port is not supported for non-parallel expander mode or VCC mode. Either the `PEXP_MODE` parameter must be set to `ON` or the `OPERATION_MODE` parameter set to something other than `VCC` if the `pterm5`port is used.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG primitive <*WYSIWYG name*> will now power up <*high or low*>

CAUSE: The specified WYSIWYG primitive was originally in a state that would power up either high or low; however, the WYSIWYG primitive will now power up in the opposite state. This condition may be caused by incorrect logic feeding the WYSIWYG's control signals, especially the preset and clear ports.

ACTION: Verify that the logic feeding the control signals in the WYSIWYG primitive is correct, especially the logic feeding the preset and clear ports.

WYSIWYG RAM primitive <*name*> cannot have <*names*> ports in mixed registered modes -- all ports must be registered or non-registered

---

CAUSE: The specified WYSIWYG RAM primitive cannot have the specified ports in mixed registered modes. The ports must all be registered or non-registered.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG RAM primitive <*name*> has <*names*> parameters with values that do not match between read and write addresses for <*A or B*> port

---

CAUSE: The specified WYSIWYG RAM primitive has conflicting values for the specified parameters between read and write addresses for the specified port. The parameter values must be changed so that they match.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG RAM primitive <*name*> has actual ADDRESS_WIDTH <*number*>, which is different from declared ADDRESS_WIDTH value <*number*>

CAUSE: The specified WYSIWYG RAM primitive has a value for the specified actual RAM `ADDRESS_WIDTH`, but the value is different from the specified `ADDRESS_WIDTH` value declared in the parameter. The actual `ADDRESS_WIDTH` parameter value must be the same as the declared `ADDRESS_WIDTH` parameter value.

ACTION: If you are using an EDA tool, please contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG RAM primitive <*name*> has actual DATA_WIDTH <*number*>, which is different from declared DATA_WIDTH <*number*>

CAUSE: The specified WYSIWYG RAM primitive has a value for the specified actual RAM `DATA_WIDTH` that is different from the specified declared `DATA_WIDTH` parameter.

ACTION: Check the design file to make sure the parameters and the logic feeding the `datain` ports have the same width value. If the parameters are correct and you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG RAM primitive <*name*> has ADDRESS_WIDTH <*number*>, which is incompatible with LOGICAL_RAM_DEPTH value <*number*> declared in parameter

---

CAUSE: The specified WYSIWYG RAM primitive has incompatible values for the specified RAM ADDRESS_WIDTH parameter and the specified LOGICAL_RAM_DEPTH parameter. The RAM ADDRESS_WIDTH and the LOGICAL_RAM_DEPTH parameter values must match.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG RAM primitive <*name*> has different clear signals feeding bits of <*name*> input bus port

---

CAUSE: The specified WYSIWYG RAM primitive has different clear signals feeding the bits of the specified input bus port. All of the bits of the port must be fed by the same clear signal.

ACTION: If you are using an EDA tool, please contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG RAM primitive <*name*> has different clock signals feeding bits of <*name*> input bus port

---

CAUSE: The specified WYSIWYG RAM primitive has different clock signals feeding the bits of the specified input bus port. All of the bits of the port must be fed by the same clock signal.

ACTION: If you are using an EDA tool, please contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG RAM primitive <*name*> has mismatched <*names*> parameters for <*name*> port and <*name*> port

---

CAUSE: The specified WYSIWYG RAM primitive has values for the specified parameters that do not match for the specified ports.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG RAM primitive <*name*> has RAM ADDRESS_WIDTH parameter value that must be in deep RAM mode

---

CAUSE: The specified WYSIWYG RAM primitive has a RAM `ADDRESS_WIDTH` value that is not in deep RAM mode. The WYSIWYG RAM primitive with a RAM `ADDRESS_WIDTH` parameter value must operate in deep RAM mode. Either the deep RAM mode must be turned on, or the RAM `ADDRESS_WIDTH` parameter value must be changed.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG RAM primitive <*name*> has too many bits

CAUSE: The number of bits in the specified WYSIWYG RAM primitive exceeds the capacity of an Embedded System Block (ESB). The `FIRST_ADDRESS`, `LAST_ADDRESS`, and `DATA_WIDTH` parameters must be modified so that the WYSIWYG RAM primitive can fit in the ESB.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG RAM primitive <*name*> must feed a single tri-state bus node

---

CAUSE: The specified deep RAM mode WYSIWYG RAM primitive does not feed a tri-state bus node. However, deep RAM mode WYSIWYG RAM primitives must feed a single tri-state bus node.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG RAM primitive <*name*> must have <*name*> parameter

CAUSE: The specified WYSIWYG RAM primitive does not have the required specified parameter. The WYSIWYG RAM primitive must have the specified parameter.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG RAM primitive <*name*> must have <*name*> port or parameter specified

---

CAUSE: The specified WYSIWYG RAM primitive does not have the specified port or a value for the specified parameter. The WYSIWYG primitive must have a connection for the specified port or a value for the specified parameter.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG RAM primitive <*name*> must have LOGICAL_RAM_DEPTH parameter

---

CAUSE: The specified WYSIWYG RAM primitive does not have a required `LOGICAL_RAM_DEPTH` parameter. A `LOGICAL_RAM_DEPTH` parameter with a legal value must be specified.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG RAM primitive <*name*> must have RAM ADDRESS_WIDTH parameter

---

CAUSE: The specified WYSIWYG RAM primitive does not have a required RAM `ADDRESS_WIDTH` parameter. A RAM `ADDRESS_WIDTH` parameter with a legal value must be specified.

ACTION: If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG RAM primitive <*name*> must have same values specified for both its WRITE_ENABLE_CLEAR and WRITE_ADDRESS_CLEAR parameters

---

CAUSE: The specified WYSIWYG RAM primitive has different values specified for its `WRITE_ENABLE_CLEAR` and `WRITE_ADDRESS_CLEAR` parameters. The WYSIWYG RAM primitive cannot be converted if the specified values for its `WRITE_ENABLE_CLEAR` and `WRITE_ADDRESS_CLEAR` parameters are different. This error can occur if you instantiated an `altdpram` megafunction in the design or if you tried to compile a Text Design Output File (**.tdo**) imported from the MAX+PLUS II software.

ACTION: If you instantiated an `altdpram` megafunction in the design, use the **MegaWizard Plug-In Manager** to make sure the settings for the `aclr` port are the same as for the `wraddress` and `wren` ports. If you attempted to compile a TDO File, use another design. If you are using an EDA tool, contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.

WYSIWYG RAM primitive <*name*> uses <*name*> port, which is inconsistent with <*name*> parameter value

---

CAUSE: The specified WYSIWYG RAM primitive uses the specified port, but that port is inconsistent with the specified parameter value. The WYSIWYG primitive must either use a parameter value that is consistent with the port, or the port must not be used.

ACTION: If you are using an EDA tool, please contact the technical support for the EDA tool regarding this message. Otherwise, contact Altera Applications for assistance.