link

NAME

link

Resolves design references.

SYNTAX

```
status link
```

ARGUMENTS

The **link** command has no arguments.

DESCRIPTION

Performs a name-based resolution of design references for the current design. For a design to be complete, it needs to be connected to all of the library components and designs it references. The references must be located and linked to the current design in order for the design to be functional. The purpose of this command is to locate all of the designs and library components referenced in the current design and connect (link) them to the current design.

The **link** command uses the **link_library** and **search_path** variables along with the **local_link_library** design attribute to resolve design references. The **local_link_library** attribute and **link_library** variable specify a list of design and library files. A "*" entry in the value of the **link_library** variable indicates that **link** should search all the designs already loaded in memory. If the **link_library** variable has no "*" entry, the already-loaded designs are not searched. The default value for the **link_library** variable is "* your_library.db". The **search_path** variable specifies a list of directory names that the **link** command uses to search for **link_library** or **local_link_library** files.

If the reference is to a parameterized design (and the design is not already in memory), **link** automatically builds the template with the specified parameters. Parameterized designs can only be specified using HDL code.

The **link** command searches for referenced designs by looking in each file specified by the **local_link_library** attribute and the **link_library** variable. If the **local_link_library** attribute is set on the current design, those files are searched ahead of those specified by the **link_library** variable. For simple file names (names that contain no '/' character), the **link** command looks for the files in the directories specified by the **search_path** variable. For absolute or relative path names, the **search_path** variable is not used. If the referenced designs are not found in any of the specified files, the **link** command looks in the search path directories for any .ddc or .db files that have the same name as the referenced design (for example, adder.ddc or adder.db). In this case, the .ddc files are searched first.

The order of directories in the search path and of the files in the link libraries is important. Search order during a link is

1. Local link library files
2. Link library files
3. Search path directories

The first occurrence of a design reference is used.

Multicorner-Multimode Support

```
This command uses information from all active scenarios.
```

EXAMPLES

In all modes, if a design named TOP refers to design named test in its implementation, the **link** command must find and connect the test design to this reference. The **link** command first searches the files specified in the link libraries for TOP. If it does not find the test design in the link libraries, it searches the directories specified in the **search_path** variable for a file named test.ddc or test.db. If it does not find it there, a warning is issued stating that the reference cannot be resolved.

Assume that the **local_link_library** attribute, **search_path** variable, and **link_library** variable for the previous example are set as follows:

```
prompt> set_app_var search_path "$synopsys_root/libraries /usr/bill/project"
/usr/synopsys/libraries /usr/bill/project

prompt> set_local_link_library tech1_lib.db
1

prompt> set_app_var link_library "* tech2_lib.db ./project.db"
* tech2_lib.db ./project.db
```

To find the test design, the **link** command performs the following sequence:

Step 1.

Read the tech1_lib.db file. Because this is a simple filename, the search path is used to find the file's location.

Step 2.

Search all the designs already loaded in memory for a design called test.

Step 3.

Read the tech2_lib.db file. Because this is a simple filename, the search path is used to find the file's location.

Step 4.

Read the project.db file from the current directory. Because this is a complex filename (the current directory is specified), the search path is not used to find the file.

Step 5.

Look for the test design in the list of designs contained in tech1_lib.db, tech2_lib.db, and project.db.

Step 6.

If the test design is not found in Step 5, search for a file named either test.ddc or test.db using the search path. In the unlikely event that test.ddc and test.db both exist, the .ddc file is used.

The following examples demonstrate the different ways the **link** command can search for referenced designs.

Example 1

The **search_path** variable is set to a list of directories where designs and technology libraries reside. In the following example, the top, counter, and controller designs are built using components from the tech_lib library. You need to read in only the top-level design. The **link** command automatically loads designs from both the link library files and, as needed, files in the search path.

```
prompt> set_app_var search_path "$synopsys_root/libraries /usr/bill"
/usr/synopsys/libraries /usr/bill

prompt> read_file top.ddc
Loading db file '/usr/synopsys/libraries/syn/gtech.db'
Loading db file '/usr/synopsys/libraries/syn/standard.sldb'
  Loading link library 'gtech'
Reading ddc file '/usr/bill/top.ddc'.
Loaded 1 design.
Current design is 'top'.
{top}

prompt> set_app_var link_library "* tech_lib.db"
* tech_lib.db

prompt> link

  Linking design 'top'
  Using the following designs and libraries:
  --------------------------------------------------------------------------
  top                           /usr/bill/top.ddc
  tech_lib (library)            /usr/synopsys/libraries/tech_lib.db

Reading ddc file '/usr/bill/controller.ddc'.
Reading ddc file '/usr/bill/counter.ddc'.
1
```

Example 2

The following example shows that reading a design into memory has no effect on the **link** command if the "*" is missing from the **link_library** variable. The adder design is read into memory explicitly, but cannot be found on the search path or in the link libraries. Therefore, it is unresolved during linking.

This command sets the **search_path** variable without including the directory that contains the adder.ddc file:

```
prompt> set_app_var search_path "$synopsys_root/libraries /usr/bill"
/usr/synopsys/libraries /usr/bill
```

The next two commands read the adder design into memory and set the current design to top:

```
prompt> read_file /usr/bill/dc/adder.ddc
Loading db file '/usr/synopsys/libraries/syn/gtech.db'
Loading db file '/usr/synopsys/libraries/syn/standard.sldb'
  Loading link library 'gtech'
Reading ddc file '/usr/bill/dc/adder.ddc'.
Loaded 3 designs.
Current design is 'adder'.
adder mux81 top

prompt> current_design top
Current design is 'top'.
{top}
```

The next command sets the **link_library** variable without including the adder.ddc file or "*":

```
prompt> set_app_var link_library "tech_lib.db"
tech_lib.db
```

The following command shows that the **link** command fails for the designs in adder.ddc:

```
prompt> link

  Linking design 'top'
  Using the following designs and libraries:
```

```
  ------------------------------------------------------------------
  tech_lib (library)           /usr/synopsys/libraries/tech_lib.db

Warning: Unable to resolve reference 'adder' in 'top'. (LINK-5)
Warning: Unable to resolve reference 'mux81' in 'top'. (LINK-5)
```

Adding "*" to the **link_library** variable allows the design to link successfully:

```
prompt> set_app_var link_library "* $link_library"
* tech_lib.db

prompt> link

  Linking design 'top'
  Using the following designs and libraries:
  ------------------------------------------------------------------
  * (3 designs)              /usr/bill/dc/adder.ddc, etc
  tech_lib (library)         /usr/synopsys/libraries/tech_lib.db

1
```

An alternate solution for the link failure is to add the adder.ddc file to the **link_library** variable:

```
prompt> set_app_var link_library "tech_lib.db /usr/bill/dc/adder.ddc"
tech_lib.db /usr/bill/dc/adder.ddc

prompt> link

  Linking design 'top'
  Using the following designs and libraries:
  ------------------------------------------------------------------
  * (3 designs)              /usr/bill/dc/adder.ddc, etc
  tech_lib (library)         /usr/synopsys/libraries/tech_lib.db

1
```

Example 3

This example shows how to handle files with multiple designs. If you need to link with a file of multiple designs, put that file in your **link_library**.

If a design is not found in the **link_library** files, **link** searches the **search_path** directories for a file of the same name as the referenced design, with a .db or .ddc suffix. If your multiple-design file has a different file name then it will not be found, even if it is located on the **search_path**.

```
prompt> set_app_var search_path "$synopsys_root/libraries /usr/project"
/usr/synopsys/libraries /usr/project
```

The following command sets **link_library** without including "mylib.ddc":

```
prompt> set_app_var link_library "tech_lib.db"
tech_lib.db
```

The link fails for designs in "mylib" that are not found in any **link_library** files or in files in **search_path** directories that are named with the design names:

```
prompt> read_file top.ddc
Loading db file '/usr/synopsys/libraries/syn/gtech.db'
Loading db file '/usr/synopsys/libraries/syn/standard.sldb'
  Loading link library 'gtech'
Reading ddc file '/usr/project/top.ddc'.
Loaded  design.
Current design is 'top'.
top

prompt> link

  Linking design 'top'
  Using the following designs and libraries:
  ------------------------------------------------------------------
  tech_lib (library)           /usr/synopsys/libraries/tech_lib.db
```

```
Warning: Unable to resolve reference 'adder' in 'top'. (LINK-5)
Warning: Unable to resolve reference 'mux81' in 'top'. (LINK-5)
```

Because the adder and mux81 designs are in the mylib.ddc file, adding this file to the **link_library** variable enables a successful link:

```
prompt> lappend link_library mylib.ddc
tech_lib.db  mylib.ddc

prompt> link
Reading ddc file '/usr/project/mylib.ddc'.

  Linking design 'top'
  Using the following designs and libraries:
  --------------------------------------------------------------------------
  tech_lib (library)           /usr/synopsys/libraries/tech_lib.db
  * (2 designs)                /usr/project/mylib.ddc, etc

1
```

SEE ALSO

current_design(2)
set_local_link_library(2)
list_designs(2)
which(2)
auto_link_options(3)
link_library(3)
search_path(3)