Attributes

Supported attributes include the following built-in attributes: *'LEFT*, *'RIGHT*, *'HIGH*, *'LOW*, *'BASE*, *'RANGE*, *'REVERSE_RANGE*, and *'LENGTH*. You can also use the attributes *'EVENT* and *'STABLE* for describing clock edges and resets only. Synergy does not support user-defined attributes. See the “Modeling Clocks” section on page 1-37 to see how to use the *'EVENT* and *'STABLE* attributes.

Synergy supports the use of attributes for setting constraints. The attributes for setting constraints map to the Synergy text user interface commands. Figure 2-34 shows the attributes you can use to set constraints.

Figure 2-34 Constraint Setting Using Attributes

Name of Attribute	Applies to Entity Class	Functionality	Attribute Type	Equivalent Text Interface Command
'PRESERVE	entity/ architecture	No instances are synthesized	*boolean*	*maintain_tree*
	instance	Contents of instance not synthesized	*boolean*	*maintain_tree*
	signal	Preserved after synthesis	*boolean*	*maintain_tree*
'PRESERVE_ HIERARCHY	entity/ architecture	Keeps boundaries; synthesizes contents	*boolean*	*preserve_boundary*
	instance	Keeps boundaries of instance	*boolean*	*preserve_boundary*
	function	Keeps boundaries of function	*boolean*	*preserve_boundary*

Figure 2-34 Constraint Setting Using Attributes, *continued*

Name of Attribute	Applies to Entity Class	Functionality	Attribute Type	Equivalent Text Interface Command
'PRESERVE_ HIERARCHY_ TREE	entity/ architecture	Applies *preserve_hier* downward in hierarchy	*boolean*	*preserve_boundary_tree*
	instance	Applies *preserve_hier* only to instance's hierarchy	*boolean*	*preserve_boundary_tree*
'EXPAND_ HIERARCHY_ TREE	entity/ architecture	Used with *preserve_hier_tree* to flatten lower levels	*boolean*	*expand_hier_tree*
	instance	Flattens a particular instance's hierarchy	*boolean*	*expand_hier_tree*
'UNROLL	labeled loop	Unrolls loop	*boolean*	*unroll*
'SYNTHESIZE	entity, instance, signal	Turns off the effect of the *preserve* command	*boolean*	*synthesize*
'MAX_AREA	entity/ architecture	Maximum area constraint	*real*	*set_max_area*
'DRIVE_FACTOR	input port, entity	Input timing constraint (to all ports)	*real*	*set_drive*
'RISE_ARRIVAL_ TIME	input port, entity	Input timing constraint (to all ports)	*real*	*set_arrv_time*
'MAX_ARRIVAL_ RISE_TIME	input port, entity	Input timing constraint (to all ports)	*real*	*set_arrv_time*
'MAX_ARRIVAL_ FALL_TIME	input port, entity	Input timing constraint (to all ports)	*real*	*set_arrv_time*

Figure 2-34 Constraint Setting Using Attributes, *continued*

Name of Attribute	Applies to Entity Class	Functionality	Attribute Type	Equivalent Text Interface Command
'FALL_ARRIVAL_ TIME	input port, entity	Input timing constraint (to all ports)	*real*	*set_arrv_time*
'MIN_ARRIVAL_ RISE_TIME	input port, entity	Input timing constraint (to all ports)	*real*	*set_arrv_time*
'MIN_ARRIVAL_ FALL_TIME	input port, entity	Input timing constraint (to all ports)	*real*	*set_arrv_time*
'LOAD	output port, entity	Output timing constraint	*real*	*set_load*
'RISE_DELAY	output port, entity	Output timing constraint	*real*	*set_req_time*
'MAX_RISE_DELAY	output port, entity	Output timing constraint	*real*	*set_req_time*
'MAX_FALL_DELAY	output port, entity	Output timing constraint	*real*	*set_req_time*
'FALL_DELAY	output port, entity	Output timing constraint	*real*	*set_req_time*
'MIN_RISE_DELAY	output port, entity	Output timing constraint	*real*	*set_req_time*
'MIN_FALL_DELAY	output port, entity	Output timing constraint	*real*	*set_req_time*
'FANIN	output port, entity	Fanin contribution from component to inputs of next stage	*real*	*set_fanin*

Figure 2-34 Constraint Setting Using Attributes, *continued*

Name of Attribute	Applies to Entity Class	Functionality	Attribute Type	Equivalent Text Interface Command
'FANOUT	input port, entity	Fanin contribution on output of previous stage	*real*	*set_fanout*
'MAX_FANOUT	input port, entity	Specifies the maximum fanout permitted on the port	*real*	*set_max_fanout*
'ASYNCHRONOUS_ SYSTEM_RESET	input port	Specifies asynchronous reset signal	*bit*	*set_sys_reset -n asynchronous*
'SYNCHRONOUS_ SYSTEM_RESET	input port	Specifies synchronous reset signal	*bit*	*set_sys_reset -n synchronous*

To set a constraint using attributes, use an attribute specification with one of the attribute names listed in Figure 2-34 as the attribute designator. Figure 2-35 shows how to specify an attribute to preserve an instance. The attribute declarations for the attributes in the table above are located in the *SYNERGY.constraints* package. To use constraint attributes in a design, include the *library* and *use* context clauses for the *SYNERGY.constraints* package as shown in Figure 2-35. See the *IEEE Standard VHDL Language Reference Manual* for the syntax of attribute specifications. See the *Synergy VHDL Synthesizer and Optimizer Reference* for a description of the text interface commands corresponding to these attributes.

If you set the same constraint attribute in both the VHDL model and the graphical or text user interface, then the constraint you set in the user interface takes precedence over the constraint in your source code.

Figure 2-35 Preserving Instances Using Attribute Constraints

Include reference to *VHS.vhs* for attribute definitions.

Use predefined attribute *ASYNCHRONOUS_ SYSTEM_RESET* to generate an asynchronous reset.

Use predefined *PRESERVE* attribute to preserve instance *i1*.

```
library IEEE;
use IEEE.std_logic_1164.all;
library VHS;
use VHS.vhs.all;
library work;

entity preserve_dff is
   port (d: in std_logic;
         clock, s, c: in std_logic;
         output: inout std_logic);
   attribute ASYNCHRONOUS_SYSTEM_RESET of c:signal is '0';
end preserve_dff;

architecture arch_preserve_dff of preserve_dff is

component dff
   port (data: in std_logic;
         clk, set, clr: in std_logic;
         outp: inout std_logic);

end component;

begin
   b1: block
      for all: dff use entity work.dff1(arch_dff1);
      attribute PRESERVE of i1: label is true;
      begin
         i1: dff port map (d, clock, s, c, output);
      end block;

end arch_preserve_dff;
```