Chapter 10
Completing Port Constraints

The delay through a specific cell depends on the slew/transition rate at its input as well as the load that it sees at its output. For cells inside the design, the fanin driver and the fanout cone is also part of the design. So the transition rate as well as the load can be computed by the tool. However, for the cells which are being driven by the input port, the input transition time is not known. Similarly, for the cells which drive the output port, the load is not known. Thus, designers need to provide the input transition time for the input signals and the external load that the output port will see. If not specified, the transition time is assumed to be *0,* namely, a sharp ramp (equivalent to infinite drive strength), and load is assumed to be *0,* namely, no external load. Both these conditions are highly optimistic.

The transition information can be specified through either of the following SDC commands:

set_drive
set_driving_cell
set_input_transition

The load information can be specified through either of the following SDC commands:

set_fanout_load
set_load
set_port_fanout_number

10.1 Drive Strength

Let us consider the circuit shown in Fig. 10.1.

Let us say the input *I1* of the block *B1* is being driven by the inverter. When the signal is transitioning to a *1,* the p-transistor of the inverter is driving the value on to the signal. This p-transistor offers some resistance to the driving signal.

S. Gangadharan and S. Churiwala, *Constraining Designs for Synthesis and Timing Analysis: A Practical Guide to Synopsys Design Constraints (SDC)*,
DOI 10.1007/978-1-4614-3269-2_10, © Springer Science+Business Media New York 2013

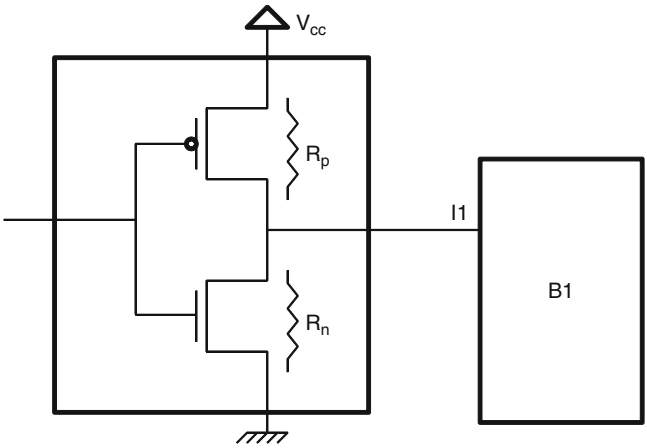

Fig. 10.1 Equivalent resistance

This resistance will influence the risetime at *I1*. When the signal is transitioning to a *0,* the n-transistor's equivalent resistor influences the fall time at *I1*.

The resistive equivalent for the p-transistor and the n-transistor might not be equal. Thus, the driver-resistance value might be different for the input transitioning to a *1* or to a *0*.

Now, let us say that the input is driven by a *NAND* gate, as shown in Fig. 10.2.

When *I1* is transitioning to a *1*, it could be because either *P1* transistor is ON, or *P2* transistor is ON, or both *P1* and *P2* transistors are ON. Depending upon which of the above situation is true, the resistive equivalent would be different. When both the transistors are ON, the resistance is minimal. Thus, the driver resistance could be within a range, rather than a specific value.

10.1.1 set_drive

The SDC command to specify the equivalent resistance of the driver is:

set_drive [-*rise*] [-*fall*]
[-*min*] [-*max*]
resistance_value port_list

It should be noted that the value provided is actually the resistive value – which is inverse of the drive strength. Higher resistance means lower drive (longer time to transition) and vice versa.

-*rise* or -*fall* is used to specify whether the drive (actually, resistance of the driver) is for the signal rising or falling. When neither -*rise* nor -*fall* is specified, the specified value is considered to be applicable for both rising input and falling input.

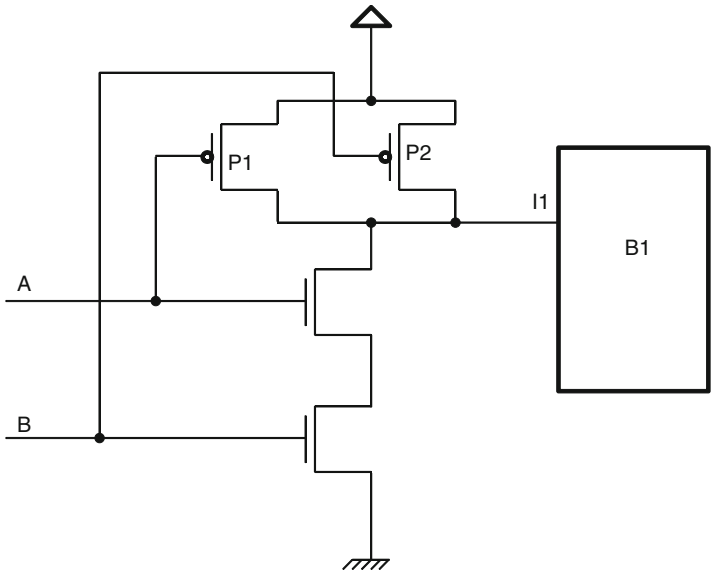

Fig. 10.2 NAND driver

-min or *-max* is used to specify whether the resistive value specified is the minimal resistance or the maximum resistance. Since minimal resistance means higher drive, hence faster transition, so minimal resistance is used for hold analysis. Similarly, max resistance is used for setup analysis. If none of *-min* or *-max* qualifiers are used, the specified value is applicable for both setup and hold analysis.

Given the resistive value of the driver, the tools can compute the slew time at the input, if they know the capacitive value of the input pin. It is worth reiterating that though the command is *set_drive*, the value specified is for resistance (which is inverse of drive).

Usually, *set_drive* is one of the less popular methods of specifying input slew.

10.2 Driving Cell

Most of the times, the internals of a cell are known only to the circuit designers for the ASIC library, rather than to the people who are using the libraries to create their designs. So it is much easier to specify the cell which will drive the input, rather than knowing the actual resistive value. The timing analysis tools have enough information from the library about the cell's electrical characteristics that they will be able to extract the relevant information.

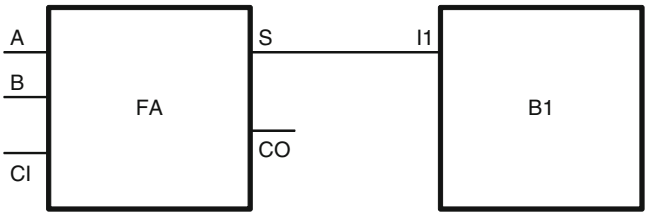

Fig. 10.3 Driver cell

10.2.1 set_driving_cell

The SDC command for specifying the driving cell is:

set_driving_cell [-*lib_cell* lib_cell_name]
[-*rise*] [-*fall*]
[-*min*] [-*max*]
[-*library* lib_name]
[-*pin* pin_name]
[-*from_pin* from_pin_name]
[-*multiply_by* factor]
[-*dont_scale*]
[-*no_design_rule*]
[-*clock* clock_name]
[-*clock_fall*]
[-*input_transition_rise* rise_time]
[-*input_transition_fall* fall_time]
port_list

10.2.1.1 Driver Cell Name

The -*lib_cell* switch is used to specify the cell which acts as the driver for the pin.

In Fig. 10.3, the input *I1* of the block *B1* is being driven by the cell *FA*. Thus, this cell should be specified as the -*lib_cell*.

Even though, SDC shows -*lib_cell* as an optional input, this switch is always found in any *set_driving_cell* command. Without this switch, the actual driver cell is not known. The rest of the command, options, or qualifiers might not have any meaning, if the driver cell itself is not known.

10.2.1.2 Min/Max, Rise/Fall

The -*rise*/-*fall* qualifier is used, when a designer wants to specify a different driving cell for a rise transition at the input pin and another cell for a fall transition at the input pin. If the qualifier is not used, then the same driver cell is used for both the

transitions. Usually, a pin is driven by the same driver cell, irrespective of whether it is transitioning to a *1* or to a *0*. These qualifiers are needed, when the input is being driven by *pull-up* or *pull-down* kind of drivers. The *pull-up* cell can be specified with *-rise* and the *pull-down* cell can be specified with *-fall*.

The *-min/-max* qualifier is used, when a designer wants to specify different driving cells for setup (max) analysis and another driving cell for hold (min) analysis. Usually, the circuit would have the same driver; hence, it might appear surprising that different cells can be specified for setup and hold analysis. However, earlier in the design cycle, a designer might not know exactly which cell will drive this pin. So a designer might want to specify the strongest of the possible set of driver cells with *-min* option and the weakest of the possible set of driver cells with *-max* option. We've already discussed that hold analysis is given much less importance; thus, most often, the driver corresponding to the setup (max) analysis is specified.

10.2.1.3 Library

Sometimes, multiple libraries might be loaded in the tool. And there might be cells with the same name in more than one library. If the specified driving cell is found in multiple libraries, the tool might use its own mechanism to decide which of these cells should be considered as the driver. The switch *-library* is used to explicitly state the library from which the driver cell should be looked up. If only one library is loaded into the tool, or if the specified driver cell name exists in only one of the loaded libraries, this switch is not needed.

Generally, multiple cells with the same name are not simultaneously loaded into a tool. Even if the cells have same functionality, but if there is some difference in their electrical parameters, they are given a different name, e.g., *AN2* (*2* input *AND* gate), *AN2H* (high-drive version of *AN2*), and *AN2LP* (low-power version of *AN2*). Thus, name clash usually does not happen for library cells. Some designers specify *-library* switch anyways – just to be explicit and ensure that only the intended cell gets specified.

10.2.1.4 Pin

For the example driver cell shown in Fig. 10.3, the driving cell has multiple outputs. Thus, the designer should specify *-pin* switch to clarify as to which of the outputs is being used to drive the input. For the circuit shown in Fig. 10.3, even if the driver (for *I1* pin of block *B1*) is known to be the *FA* cell, the drive strength of its output pins *S* and *CO* would be different. So the designer should mention the pin which drives the input (namely, pin *S* for the example given).

10.2.1.5 Timing Arc

A given output pin (either a cell with single output pin or for a multi-output cell – the pin specified with *-pin* switch) would have multiple timing arcs.

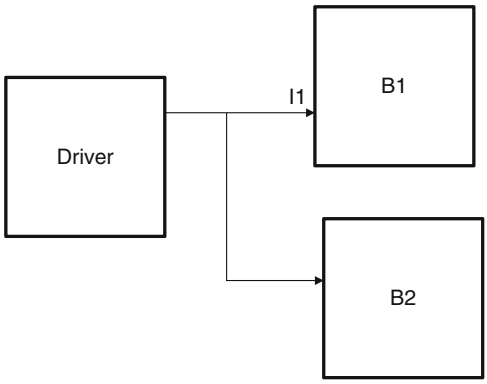

Fig. 10.4 Driver with multiple loads

And the transition time on the output pin would depend on the timing arc that is chosen. *-from_pin* switch is used to specify the input pin of the driver cell, from which the arc should be chosen. So if the designer wants to specify that even for the *S* pin of the driver cell *FA*, the arc chosen should be the *A to S* path, then the *-from_pin* should specify *A*.

The switch may be useful when a designer wants to choose an extreme condition of timing and might want to specify the arc. Typically, if *S* pin is driving the input *I1*, then all transitions on *S* will reach the input *I1*.

10.2.1.6 Multiplication

The number specified with this option is the factor by which the computed transition time gets multiplied. Effectively, it specifies the factor by which the drive strength is considered to be reduced for the driver cell. Let us consider the circuit shown in Fig. 10.4.

For the input pin *I1* of the block *B1*, the driver cell can be specified. If needed, other option *(-library, -pin, -from_pin)* may also be specified in order to more accurately control the specific arc. However, the entire drive strength is not used to drive this input pin (*I1*). The drive strength is used to drive another pin (on *B2)*. So the effective drive strength applicable for *I1* pin of *B1* has to be divided. This effect can be achieved through the switch *-multiply_by*.

However, this effect is usually achieved through *set_load* command – as we will see later in Sect. 10.6.5 of this chapter.

10.2.1.7 Scaling

Certain electrical characteristics might need to be scaled based on the operating conditions. Another word for this scaling is *derating*. If you don't want the characteristics to be scaled or derated, you can use the switch *-dont_scale*.

10.2.1.8 Design Rules

If the driver pin has some design rule properties (e.g., the highest load that it can drive), those properties get transferred to the input port. The switch *-no_design_rule* prevents the properties from getting transferred to the input port. Any block boundary is only for use in modeling. In the realized circuit, all hierarchies would finally get dissolved. The signals being driven by the input port would finally get driven by the driver pin. Thus, all design rules applicable for the driver pin should be honored by the input port also, i.e., the design rules for the driver pin should get transferred on the input port.

10.2.1.9 Clock Association

If you want the driving cell to be specified only with respect to those *set_input_delay* which is specified with respect to a specific clock, the clock association can also be specified. Let us consider the following examples:

set_input_delay -clock CLK1 3.0 [get_ports I1]
set_input_delay -clock CLK2 4.2 [get_ports I1] -add_delay

Now, if we specify:

set_driving_cell BUF1 [get_ports I1]

it would mean that *BUF1* will be used as a driver for both the above input_delay specifications.

However, if we were to specify:

set_driving_cell BUF1 -clock CLK1 [get_ports I1]

that would mean that the driving cell, *BUF1*, would be used only for the first input_delay specification.

So the clock association can be used to limit the driving cell specification only for certain input_delay (those associated with the same clock).

Even if an input has delays specified with respect to multiple clocks, the pin would still be finally driven by a single cell. However, sometimes this option is used in conjunction with a few other options. Let us consider the circuit shown in Fig. 10.5.

For the input pin *I1* for the block *B1*, *set_input_delay* would be specified with respect to clocks *CLK1* and *CLK2*. A very accurate modeling of driving cell can be done using the following set of commands:

set_driving_cell -lib_cell MUX21 -from_pin A -clock CLK1 [get_ports I1]
set_driving_cell -lib_cell MUX21 -from_pin B -clock CLK2 [get_ports I1]

Thus, for the paths originating from *CLK1*, the *A to Z* arc (of *MUX21*) gets used for the driving cell, and for the paths originating from *CLK2*, the *B to Z* arc gets used for the driving cell.

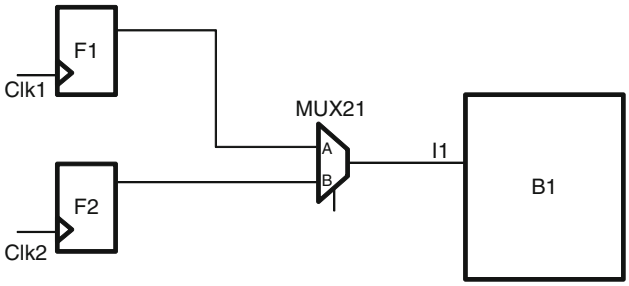

Fig. 10.5 Clock specification for driving cell

This is an example where even though some options individually do not make much sense, however, in combination with some other options, they are able to provide a much more accurate modeling.

10.2.1.10 Input Transition

These options are used to specify the rise and fall transition times at the input of the timing arc of the driver cell. Looking again at the circuit in Fig. 10.5, we could use *-input_transition_rise* and *-input_transition_fall* to specify the rise and fall transition values at the inputs *A* or *B* for the driver cell, *MUX21*. This rise and fall transition in turn will impact the transition time at the output *Z* of the *MUX21*, which will be seen at the input of the block *B1*.

It should be noticed that these transition times are not at the input of the block under consideration. Rather, these are the transition times at the input of the driving cell, which itself lies outside the block of interest. Clearly, this is second order of accuracy.

10.2.1.11 Ports

The designer has to specify the list of ports for which the driving cell property is being applied.

The most commonly used options for the *set_driving_cell* command are the name of the driving cell and the port for which the driving cell is specified. All other switches and options are used much less frequently. These other options are used under some specific situations, in order to achieve much finer control – as explained in above sections.

10.3 Input Transition

The commands *set_drive* and *set_driving_cell* are used by the tools to compute the transition time at the input port. However, a designer could specify the transition time directly. The SDC command for specifying the input transition time directly is:

set_input_transition [-*rise*] [-*fall*]
[-*min*] [-*max*]
[-*clock* clock_name]
[-*clock_fall*]
transition port_list

The significance of -*rise*, -*fall*, -*min*, and -*max* qualifiers and the clock specifications are the same as mentioned for *set_driving_cell*. They are not being explained in this section to avoid repetition. The fundamental difference with respect to *set_driving_cell* is that usually, the driver cell remains unchanged, and hence, these qualifiers are not used that often with *set_driving_cell*. However, transition times are different for rise and fall or for min and max. Hence, *set_input_transition* often uses these qualifiers to specify different transition times for rise and fall and for setup and hold analysis.

The transition values specified are the actual transition times for the specified ports.

10.3.1 Input Transition Versus Clock Transition

In Chap. 8, we had seen *set_clock_transition* command. The main difference between *set_clock_transition* and *set_input_transition* is that for the transition time specified with the *set_clock_transition*, the specified value is used for the entire clock network, rather than computed for different points on the network, while the transition time specified with the *set_input_transition* is used only for the specified port. For all other points in the fanout cone, the transition time is computed. Let us consider the circuit shown in Fig. 10.6.

If the transition is specified on *clk* port using *set_clock_transition*, then the same transition value will be used at the clock terminals of all the flops in the fanout cone of the *clk* port. For all the nets in the network, namely, *n1, n2, n3*, …, the same transition time would be assumed to be applicable.

However, if the transition time is specified on *clk* port using *set_input_transition*, then the transition rate would be computed at each of the points, namely, *n1, n2, n3*, … – including the clock terminals of all the flops.

A good usage of these two constructs is: Before the clock tree is routed, the clock net could be driving a huge fanout. Trying to compute the transition time on the clock net could result in a very very slow transitioning signal – due to a heavy load. So, in such cases, *set_clock_transition* should be used. It is expected that the clock tree synthesis will ensure that the transition rate on the clock network remains

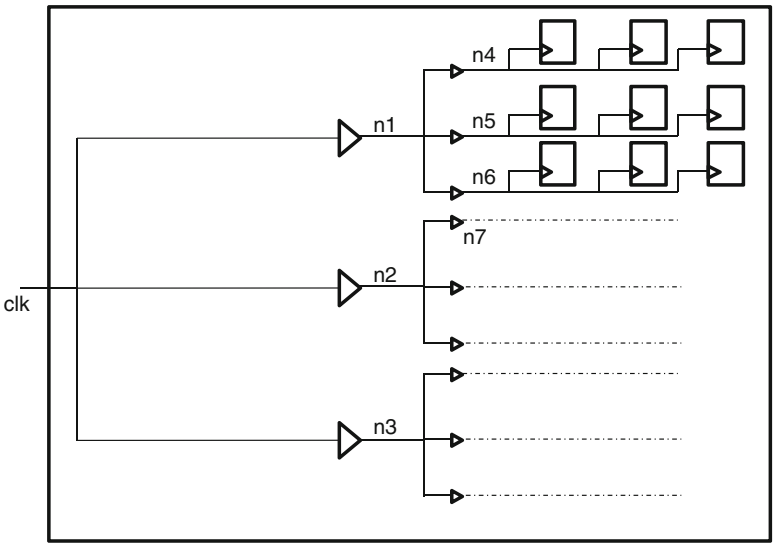

Fig. 10.6 Clock network

within a reasonable value. Once the clock tree has been synthesized, it is better to actually compute the transition value at all points. Use of *set_input_transition* (instead of *set_clock_transition*) at this stage will cause the timing analysis to compute the real transition value in the fanout cone of the *clk* port.

10.4 Fanout Number

Many wire-load models depend on the number of fanout pins to estimate the effective wire capacitance. The SDC command to specify the fanout number is:

set_port_fanout_number value port_list

This is a very simple command that specifies the fanout count of various output ports. This command has no implication, if external parasitic load is known and is being specified (through *set_load* command explained in Sect. 10.6). Since the value is the number of pins in the fanout of the port, it is expected to be an integer.

10.5 Fanout Load

Let us consider the circuit shown in Fig. 10.7.

The output *O1* drives two pins, a buffer and an *AND* gate. However, the load exhibited by the buffer is different than the load exhibited by the *AND* gate. The fanout number as mentioned in the previous section just considers the number

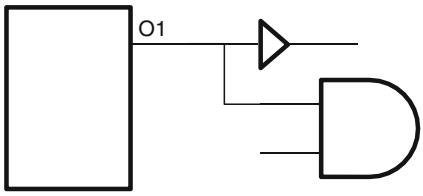

Fig. 10.7 Fanout load

of pins being driven. It does not give any idea of the effective loading that those pins together exhibit. The output port needs to know the total effective load that it sees. SDC allows designers to specify the total load in terms of multiple of standard fanout loads. The standard load is defined in the library. The SDC command to specify the external load in terms of standard load is:

set_fanout_load value port_list

Like *set_port_fanout_number*, this is also a simple command. Let us consider that the load exhibited by the buffer is one standard load. And also assume that the load of the *AND* gate is *1.5* times the load of the buffer. So AND gate's load becomes *1.5* standard load. Thus, the load seen by the port *O1* is *2.5* (standard load). This example should also explain the fundamental difference between *set_fanout_load* and *set_port_fanout_number*. For the same circuit, the *set_port_fanout_number* would be *2*, because the port drives *2* pins.

10.6 Load

When external load is expressed in terms of standard load, the tools convert the fanout values into the equivalent capacitive load. A more commonly used style of expressing external load is by specifying the external capacitance value directly, rather than the fanout. The SDC syntax for *set_load* command is:

set_load [*-min*] [*-max*]
[*-subtract_pin_load*]
[*-pin_load*]
[*-wire_load]*
value objects

The *-min/-max* qualifiers have the usual meaning. The *-min* value is to be used during hold analysis, and the *-max* value is to be used during setup analysis.

10.6.1 Net Capacitance

One of the most important things to note is that *set_load* can be applied even on wires, which are internal to the design under analysis. Thus, it provides a very convenient method to annotate extracted net capacitance obtained after post-layout.

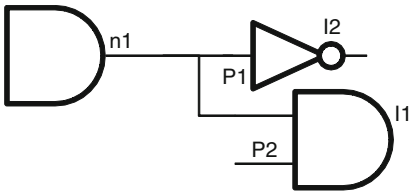

Fig. 10.8 Pin load adjustments for net capacitance

After the layout is done, effective net capacitance of each net can be extracted. And for the timing analysis tools, the capacitance value can be annotated on each net, through the *set_load* command. This allows net capacitance to be extracted by tools which are more accurate in extraction and used by STA tools.

10.6.2 Pin Load Adjustments

Let us consider the circuit shown in Fig. 10.8.

Let us say that during extraction of net capacitance for net *n1*, the extraction tool also included the loading of the pins *I1/P2* and *I2/P1*. Now, this capacitive value gets annotated on the net *n1*. The timing analysis tool sees the load on net *n1* which it considers to be the wire load only and then adds the load due to pins, *I1/P2* and *I2/P1*. So the load due to the pins gets counted twice.

So, in order to avoid double-counting, the switch *-subtract_pin_load* needs to be specified while annotating net capacitance. However, this switch should be specified only if the pin capacitances were also included during extraction. Most extraction tools will extract the net capacitance separately. If only that net capacitance is being annotated, this switch should not be specified.

Designers need to understand their extraction methodology, before deciding whether or not the pin load adjustments have to be made when specifying load on nets.

10.6.3 Load Type

Whether the specified load is of type pin or wire is specified using switches *-pin_load* or *-wire_load*. Tools might treat wire and pin loads differently. For example, slew might be degraded when propagating through wires. Let us consider the circuit in Fig. 10.9.

The slew *s1* at the output of the *AND* gate is computed based on the characteristics of the *AND* gate, the slew at the corresponding input pin of the gate and the load at the gate output. This signal then moves across the wire to the input of the buffer. However, as the signal travels across the wire, the slew characteristics get changed by the time it reaches the input of the buffer, so that the slew *s2* at the buffer input is

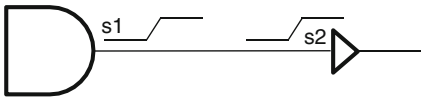

Fig. 10.9 Slew degradation through wire

different from the original slew, *s1*. This concept called slew degradation occurs for wires and not for pins. Hence, it is important to convey whether the specified capacitive load is for net or for pin.

10.6.4 Load Versus Fanout Load

set_load and *set_fanout_load* differ in the sense that *set_load* specifies the actual capacitive value of the load, while *set_fanout_load* specifies the load in terms of a standard load.

Capacitive load (specified by *set_load*) = standard loads (specified by *set_fanout_load*) × capacitive load of single standard load .

During earlier days, standard load used to be more commonly used mechanism to specify pin loads; however, standard loads are not used that commonly in current technology libraries.

10.6.5 Load at Input

Usually, load is specified at output ports. However, sometimes load might need to be specified at input ports also. Let us revisit the circuit in Fig. 10.4.

For the block *B1*'s *I1* pin, if drive strength or driver is specified and the same driver is seeing additional load, then the effective drive available to *I1* pin is reduced. That additional load needs to be specified as a load on the input pin *I1* so that the effective drive can be adjusted accordingly.

This load needs to be specified only if the drive at *I1* is being specified as drive strength (*set_drive*) or a driving cell (*set_driving_cell*). If an input transition is specified for *I1*, the load at the input does not have any impact.

10.7 Conclusion

With the transition times and the load values specified, the inputs and outputs are fully constrained. The bidirects need to be constrained as if they are both inputs (i.e., input transition) and outputs (i.e., load). Usually, clocks or reset pins are driven by higher drive cells. Thus, drive strengths for clock/reset ports/pins are usually different from the drive strengths for other functional pins.

Though there are several different commands, the tools actually need input transition time and the output load. If the information is provided in another form (e.g., drive, driving cell, standard load), then the information is transformed into input transition and output load.

Typically, during the early stage of a design, the actual details of the driver are not known, as all the blocks are being built bottom-up; at this stage, it is better to constraint the input transition through commands like *set_input_transition*. As the design progresses and various modules and the top level SoC are synthesized and available, the actual driver cell is known. At this stage, the actual drivers and arcs can be specified using *set_driving_cell*.

For output specification, the more commonly used command is *set_load*.