What Are Constraints?

Constraints provide specifications that the design must meet through optimization.

Typical examples of constraints are:

- Clock constraints
- External constraints
- Power constraints
- Net Delay constraints
- Environmental constraints
- Design rules for manufacturing

Constraint Formats

The Standard Design Constraints (SDC) format is the standard for writing constraints in the industry.

STA tools also have their own style of writing constraints, which conform to Tcl syntax.

SDC Format Equivalence

As you look at examples of constraints, you might notice slight differences in format.

The following list shows the three equivalent ways to create SDC constraints:

- Manually written constraint

  ```
  create_clock -period 100 -waveform {0 50} clk
  ```

- Generated constraints from other tools:

  ```
  create_clock -period 100 -waveform {0 50} [get_ports {clk}]
  create_clock -period 100.000000 -waveform {0.000000 50.000000} [get_ports {clk}]
  ```

All three of the above formats are functionally equivalent. Some tools might prefer one way to another

- Manually written constraint

  ```
  create_clock -period 100 -waveform {0 50} clk
  ```

- Generated constraints from other tools:

  ```
  create_clock -period 100 -waveform {0 50} [get_ports {clk}]
  create_clock -period 100.000000 -waveform {0.000000 50.000000} [get_ports {clk}]
  ```

Common SDC Constraints

Operating conditions

- set_operating_conditions

Wire-load models

- set_wire_load_mode
- set_wire_load_model
- set_wire_load_selection_group

Environmental

- set_drive
- set_driving_cell
- set_load
- set_fanout_load
- set_input_transition
- set_port_fanout_number

Design rules

- set_max_capacitance
- set_max_fanout
- set_max_transition

Timing

- create_clock
- create_generated_clock
- set_clock_latency
- set_clock_transition
- set_disable_timing
- set_propagated_clock
- set_clock_uncertainty
- set_input_delay
- set_output_delay

Exceptions

  - set_false_path
  - set_max_delay
  - set_multicycle_path

Power

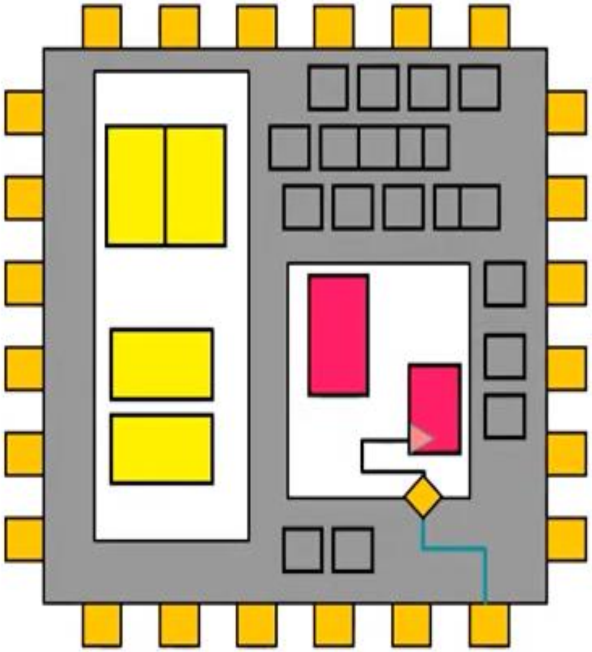
- set_max_dynamic_power
- set_max_leakage_power

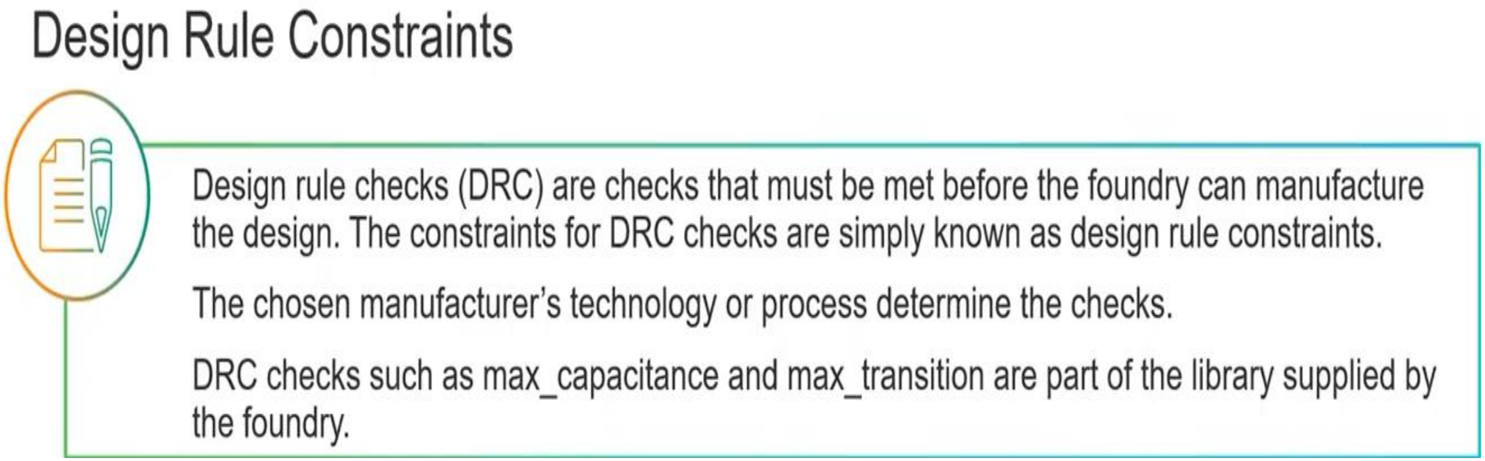
Design Objects

Object	Command	Description
Design	current_design	Design is a container for cells or is the entire circuit.
Cell or Block	get_cells	Cell is an instance of a design or is a library component.
Port	get_ports all_inputs all_outputs	A port is a signal entry point or exit point to a design.
Pin	get_pins	A pin is a hierarchical port of a design, port of an instance, or a port of a library cell.
Clock	get_clocks all_clocks	A clock is a port or a pin that drives sequential cells.
Net	get_nets	A net is an interconnect between cell pins and design ports.

You apply certain constraints to design objects to affect different parts of the design.

The table shows several design objects and the commands to get a list of these objects.





Design Rule Constraints

Design rule checks (DRC) are checks that must be met before the foundry can manufacture the design. The constraints for DRC checks are simply known as design rule constraints.

The chosen manufacturer’s technology or process determine the checks.

DRC checks such as max_capacitance and max_transition are part of the library supplied by the foundry.

Design rule constraints can be applied at the design level.

- You cannot relax these constraints, because they are supplied by the foundry to meet certain specifications.
- A more restrictive setting can be applied to specific pins, if you choose.

The following commands set these design rule constraints in SDC:

- set_max_capacitance <design | output pins>
- set_max_fanout <design | output pins>
- set_max_transition <design | input pins>

Setting Operating Conditions

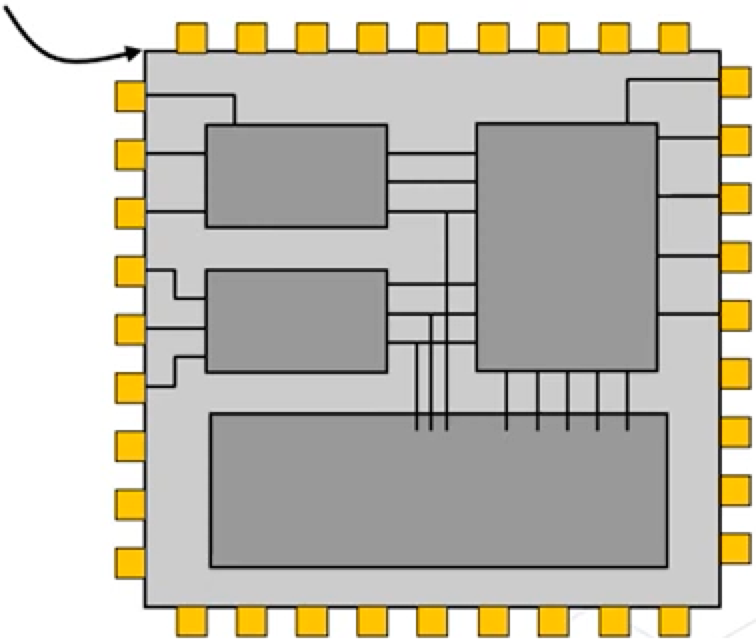
To switch between the operating conditions use:

```
set_operating_conditions
```

Operating conditions are applied at the design level. Nowadays, the technology library is created for a specific set of operating conditions. Thus, specifying the library is probably sufficient.

```
set_operating_conditions -name
  “WCCOM” -library “slowtech”
```

The timing analysis tool resets itself when you switch between operating conditions, and creates different timing data based on the new library information.



set_operating_conditions

Setting Wire-Load Mode: Top

To set the wire-load mode to *top*, use:

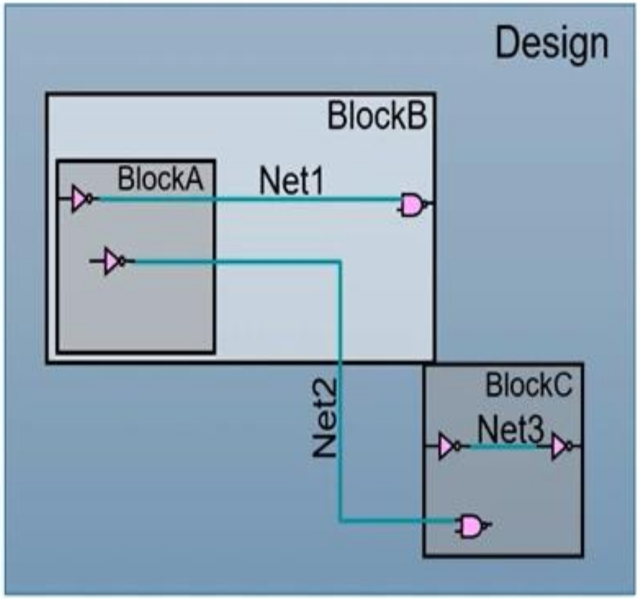
```
set_wire_load_mode top
```

The wire-load mode is applied at the design level.

- You can choose *top*, *enclosed*, or *segmented* modes.

In *top* mode, all the net delays are based on the entire design area.

If the default wire-load model in the library are based on a zero WLM, all the net delays will be 0. This approach is used in synthesis to check constraint sanity.



In **top** mode, all the nets in the design use the *Large_wlm* model.

Setting Wire-Load Mode: Enclosed

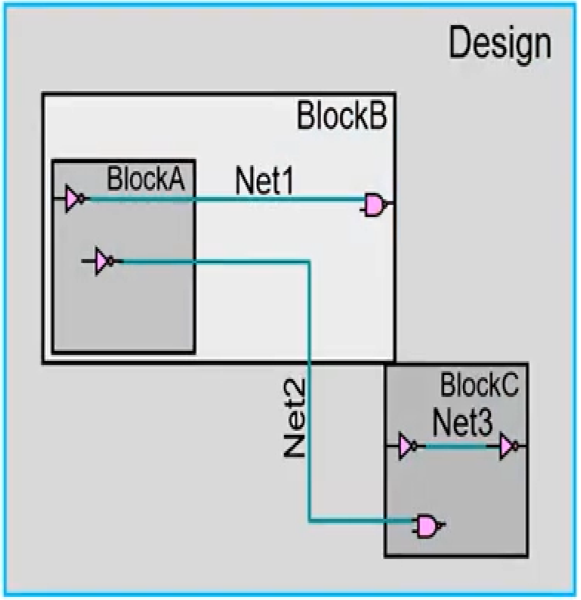
To set the wire-load mode to *enclosed*, use:

```
set_wire_load_mode enclosed
```

In *enclosed* mode, net delays are based on different block sizes, rather than on the entire design area.

In this illustration, the wire-load models for the blocks are chosen automatically from the library as follows:

- Nets enclosed in Design, use *Large_wlm*.
- Nets enclosed in BlockB, use *Avg_wlm*.
- Nets enclosed in BlockA and BlockC, use *Small_wlm*.



In **enclosed** mode:

Net1 uses *Avg_wlm*

Net2 uses *Large_wlm*

Net3 uses *Small_wlm*

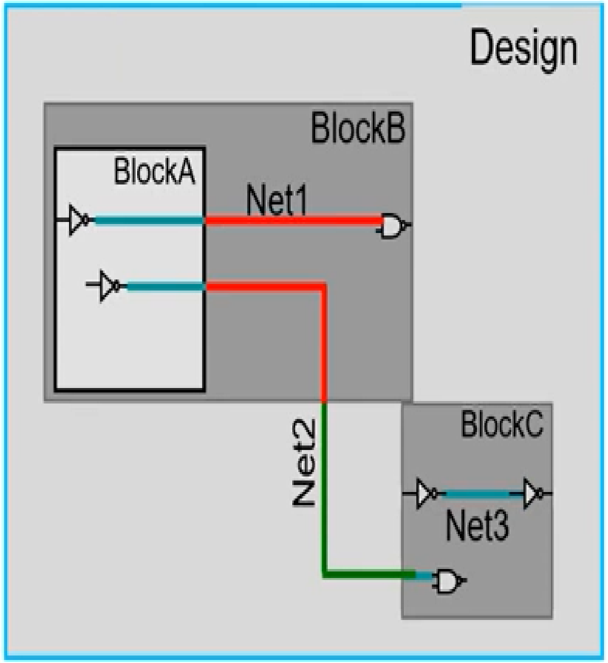
Setting Wire-Load Mode: Segmented

To set the wire-load mode to *segmented*, use:

```
set_wire_load_mode segmented
```

In *segmented* mode, the tool adds the delays from each net segment to get the total net delay. In this illustration, the wire-load models for the blocks are chosen automatically from the library as follows:

- Nets segments in Design, use *Large_wlm*.
- Nets segments in BlockB, use *Avg_wlm*.
- Nets segments in BlockA and BlockC, use *Small_wlm*.



In **segmented** mode:

Segment of Net1 in BlockA uses Small_wlm

Segment of Net1 in BlockB uses Avg_wlm

All of Net3 uses Small_wlm

BlockA/Net2 uses Small_wlm

BlockB/Net2 uses Avg_wlm

Design/Net2 uses Large_wlm

BlockC/Net2 uses Small_wlm

Setting Wire-Load Models

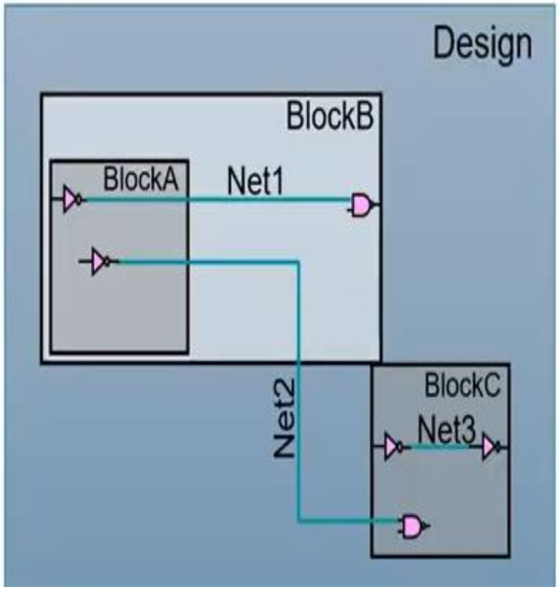
To force a WLM on a block, use:

```
set_wire_load_model
```

Wire-load models are applied at the block or design level. The STA tool automatically chooses the default wire-load model (WLM) from the library to determine the net delays.

Use these set_wire_load_model commands for the design example on the right, chosen based on the size of the block.

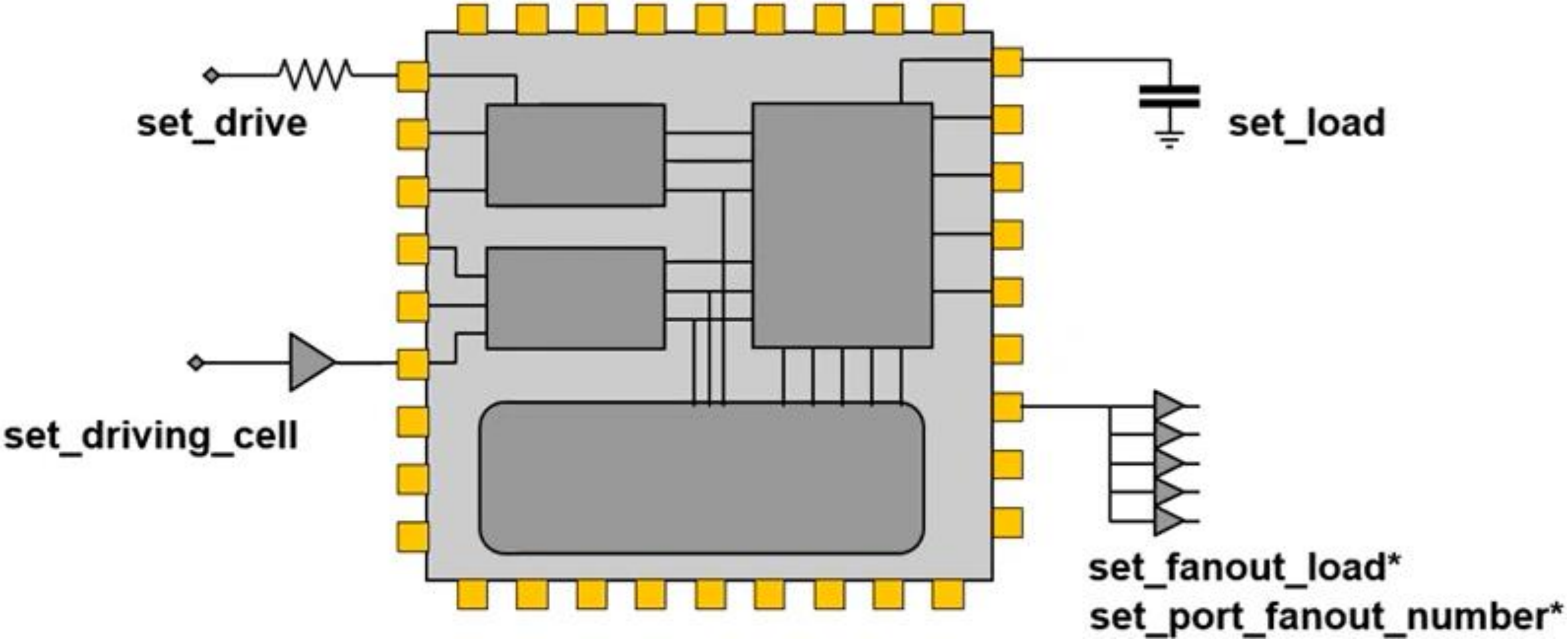
- `set_wire_load_model -name Large_wlm {Design}`
- `set_wire_load_model -name Avg_wlm [get_cell {BlockB}]`
- `set_wire_load_model -name Small_wlm [get_cell {BlockA BlockC}]`



Setting Environmental Constraints

The environment of the chip has an influence on chip performance. Environmental constraints are applied at the port level.

You need to model the environment in terms of constraints:

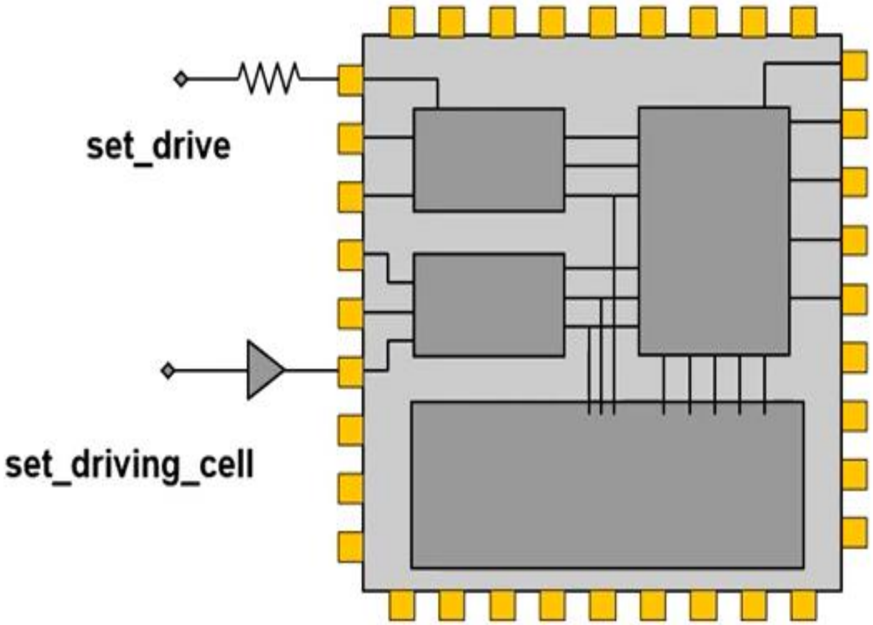


Setting the Driving Cell

Use the *set_driving_cell* command to define the cell that drives the input port.

```
set_driving_cell -lib_cell BUFX8_TAX0  -library xlite_core [get_ports
   {ipmitxbfr_rdata[3]}]
```

The input drive of a pin is the strength of incoming signal, or its capacity to drive the input and its fanout.

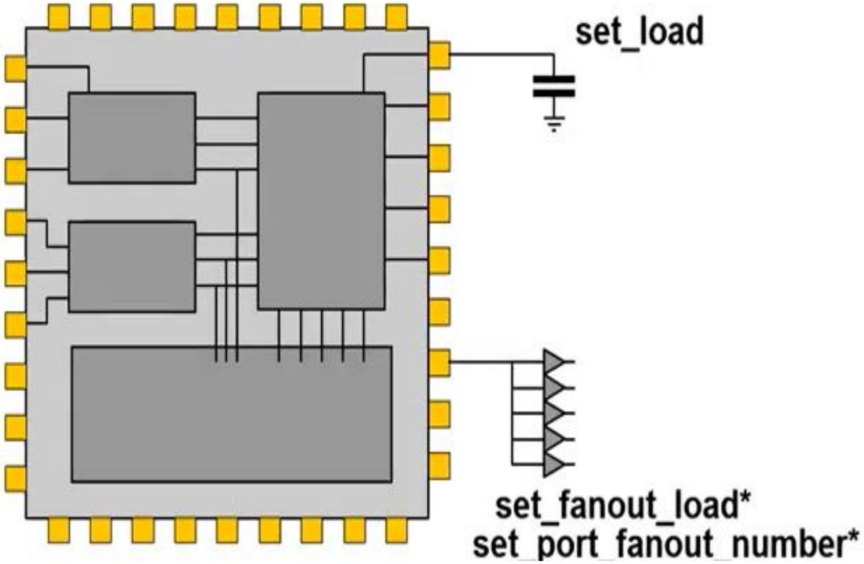


Setting Output Load

The *set_load* command lets you define the net capacitance load that the output pin "sees."

```
set_load -pin_load 0.2 [get_ports {decalf_hdrmem_ld}]
```

The load of the output port is the amount of capacitance that the port has to drive. It is the total capacitance as seen by the output port. If multiple pins are driven by the output port, then it is a summation of all the capacitances of individual endpoints.



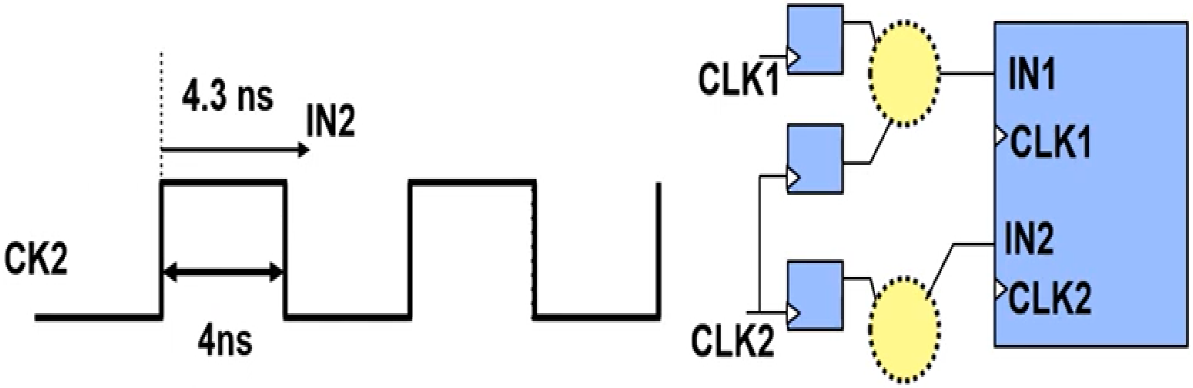
Setting Input Delay

Use the set_input_delay command to set the arrival time on a port relative to a clock.

```
set_input_delay delay [-clock clock][-add_delay] port_pin_list
```

```
set_input_delay 4.3 -rise -clock CLK2 {IN2}
```

The command in the example implies that the input signal arrives at the *IN2* port, 4.3 ns after the active edge of the *CLK2* clock.

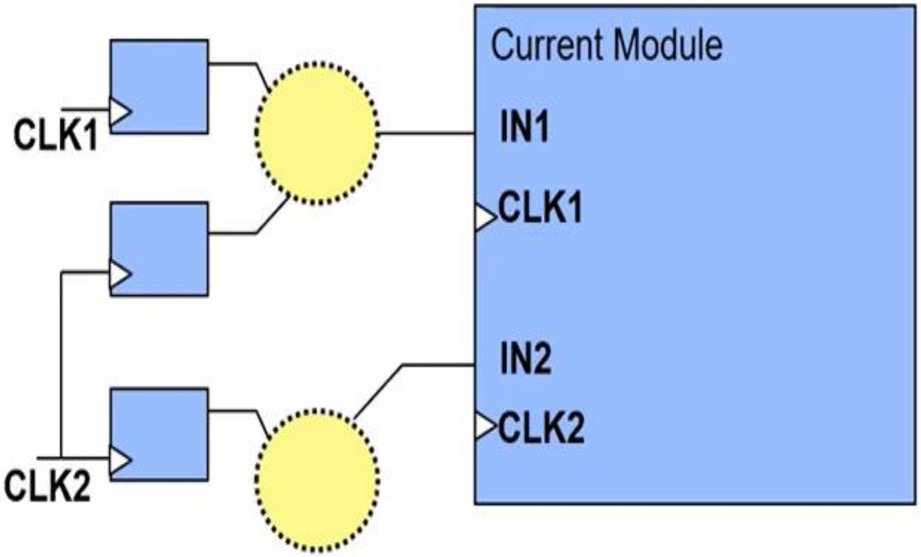


Setting the Input Delay on Ports with Multiple Clock Relationships

Use *-add_delay* to capture multiclock delay relationships.

```
set_input_delay 2.7 -clock CLK1 -add_delay { IN1 }
set_input_delay 4.2 -clock CLK2 -add_delay { IN1 }
```

The command in this multiclock example specifies an input delay for IN1, of 2.7 ns relative to CLK1 clock, and 4.2 ns relative to CLK2.



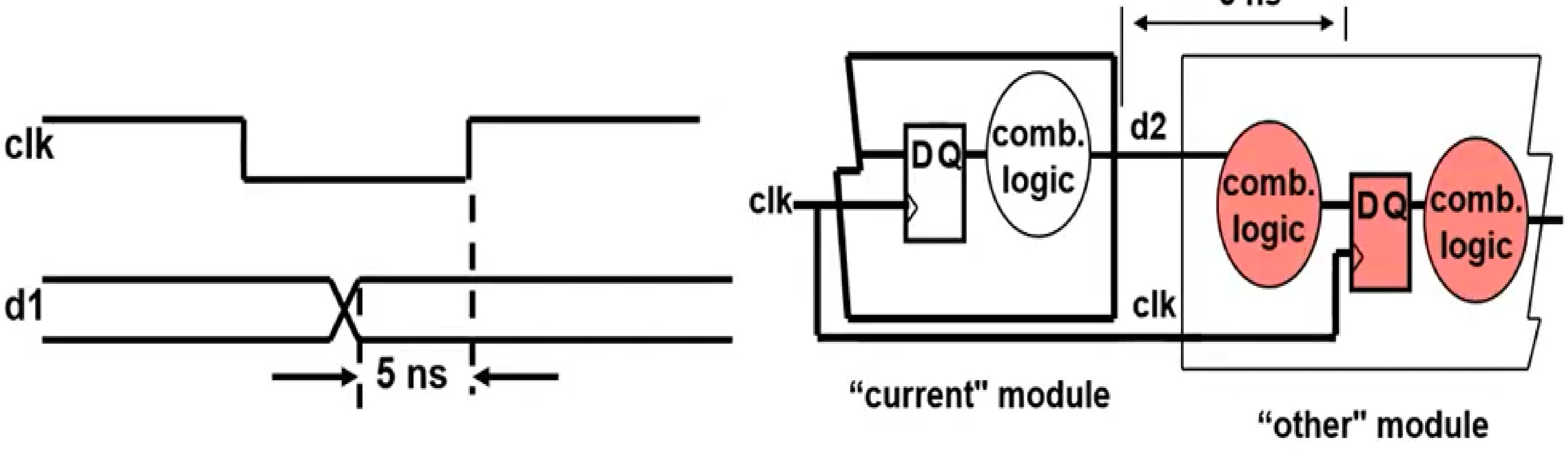
Setting Output Delay

Use the *set_output_delay* command to set the output path delay values for the design.

```
set_output_delay delay [-clock clock][-add_delay] port_pin_list
```

```
set_output_delay 5 -rise -clock clk { d1 }
```

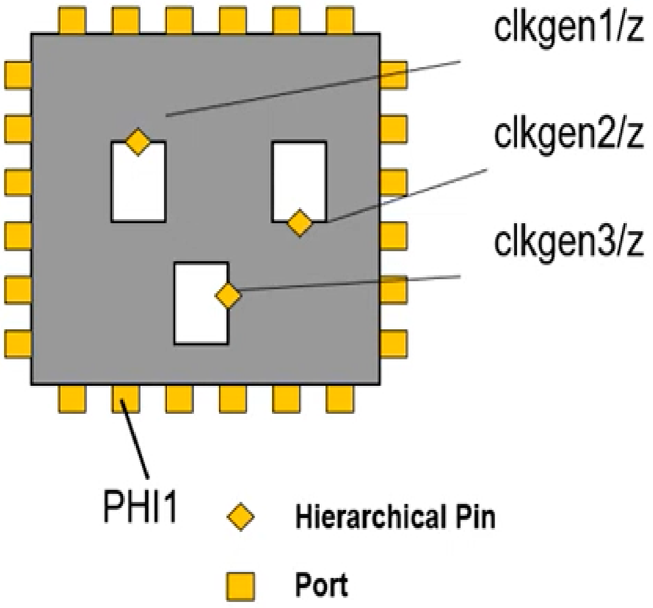
The command in the example sets an expected output delay, of **5 ns** relative to clock *clk*.



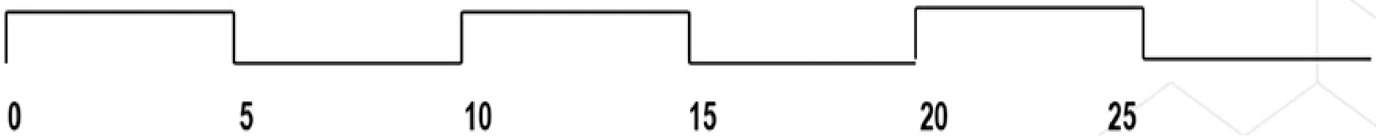
Creating a Clock

To define an clock signal and specifies the ports or pins that are connected to the clock, use:

```
create_clock -name -period -waveform {rising_edge  falling_edge} \
      <port/pin>
```



```
create_clock -name “clock1” -period 10 -
  waveform {0.0 5.0} {PHI1}
create_clock -name "clock2" -period 20 -
  waveform  {0.0 10.0} {clkgen1/Z clkgen2/Z
  clkgen3/Z}
```



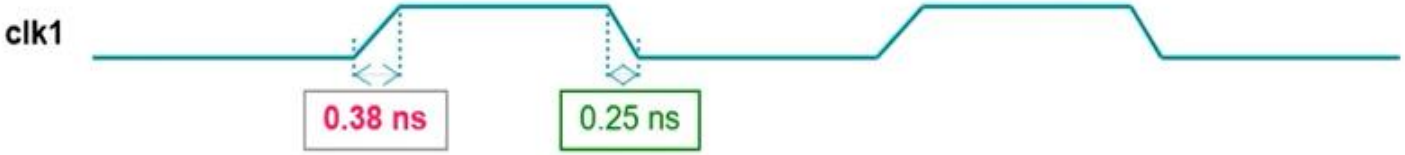
Setting Clock Transition

Clock transition is also known as clock slew. It is set on an ideal clock using the *set_clock_transition* command.

```
set_clock_transition [-rise] [-fall] [-min] [-max] \
  transition_value clock_list
```

```
set_clock_transition 0.38 -rise [get_clocks clk1]

set_clock_transition 0.25 -fall [get_clocks clk1]
```



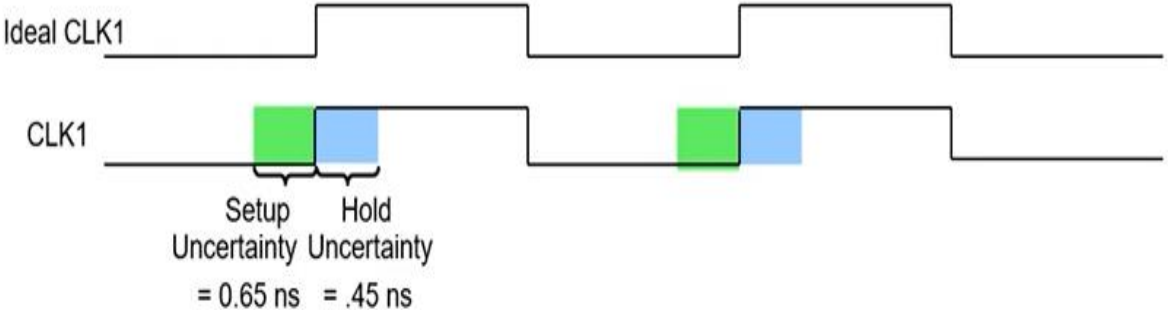
Setting Clock Uncertainty

Use the *set_clock_uncertainty* command to account for additional margin for setup and hold.

```
set_clock_uncertainty [-from from_clock] [-to to_clock] [-rise] [-fall] \
  [-setup] [-hold] uncertainty [clock_objects]
```

```
create_clock -name CLK1 -period 10 -waveform { 0 4} [get_ports CLK1]
set_clock_uncertainty -setup 0.65 [get_clocks CLK1]
set_clock_uncertainty -hold 0.45 [get_clocks CLK1]
```

The above commands specify a setup uncertainty of 0.65 ns, and a hold uncertainty of 0.45 ns on the CLK1 clock.



Setting Clock Latency

Use the *set_clock_latency* command to define clock latency.

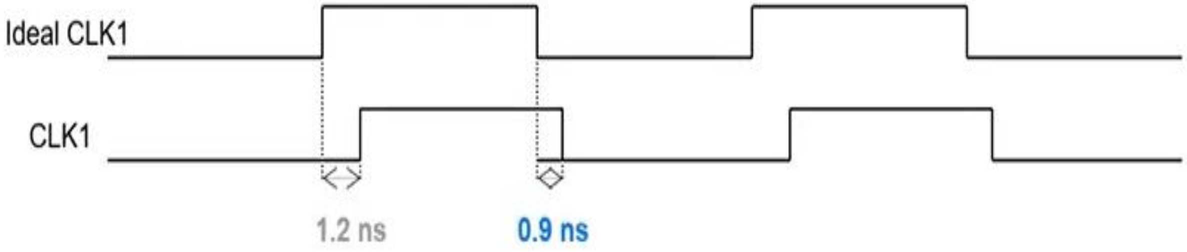
```
set_clock_latency [rise] [fall][-min] [max]
   [source] [late][early] delay object_list
```

In addition to clock slew and clock uncertainty, *clock latency* is set on the clock port to account for the delays in the clock signal from the clock definition point to a register clock pin.

Example

```
set_clock_latency 1.2 rise [get_clocks CLK1]

set_clock_latency 0.9 fall [get_clocks CLK1]
```



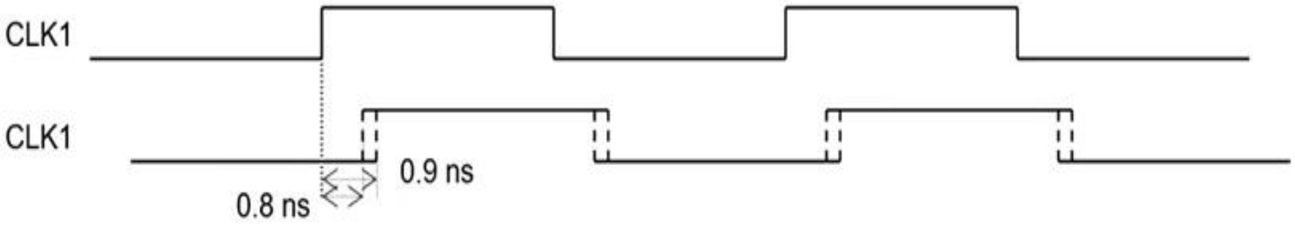
Setting Clock Latency: Hold and Setup

Use *-early,* and *-late* clock latency settings for hold and setup checks respectively.

```
set_clock_latency 0.8 -source -early [get_clocks CLK1]

set_clock_latency 0.9 -source -late [get_clocks CLK1]
```

The example commands above define a *clock source latency* of 0.8 for hold and 0.9 for setup.


CLK1
CLK1
0.9 ns
0.8 ns