Constraint Taxonomy

Version 0.6

Design Constraints Working Group

Open Verilog International

Virtual Socket Interface Alliance

Revision History

Revision	Date	Name	Comments
0.6	2/1/99	Mark Hahn	Converted to Word 97, added new clock

			constraints
0.5	10/26/98	Mark Hahn	Split into master, subdocuments
0.4	10/12/98	Mark Hahn, Greg Schulte, Steve Grout	Revisions to clock constraints, timing exceptions, and boundary parasitics
0.3	8/24/98	Mark Hahn	Merged input from Jin-sheng Shyr, Steve Grout, and Greg Schulte
0.2	7/6/98	Mark Hahn	Added a table of contents, clock constraints, and did some reformatting
0.1	6/8/98	Mark Hahn	Initial Creation

Table of Contents

1. Timing Domain

1.1 Clock Constraints

Clock

Specifies a set of points in the design at which a clock waveform originates

Type Environment condition

Applies To Clock roots (hierarchical pins or primitive outputs)

Depends On Waveform, Derived Waveform

Verified By None

Parameters

Name	Value Type	Restrictions	Semantics
Waveform	String	Waveform can only have two edges	The name of the abstract waveform

Derived Clock

Specifies a parent clock root and one or more derived clock roots. Phase shift is calculated automatically between the parent clock root and the derived clock roots.

Type Environment condition

Applies To Derived clock roots (hierarchical pins or primitive outputs)

Depends On Clock, Waveform, Derived Waveform

Verified By None

Parameters

Name	Value Type	Restrictions	Semantics
Parent clock root	Pin name		Point at which to start calculating phase shift
Waveform	String	Waveform can only have two edges	The name of the abstract waveform

Clock Arrival

Specifies insertion delay in the portion of a clock network that lies outside of the current design, where that insertion delay affects the relative arrival time of the clock edges at clock roots within the design.

Type Environment condition

Applies To Clock roots (hierarchical pins or primitive outputs), waveforms, or clock leaves

Depends On Clock, Derived Clock

Verified By None

Parameters

Name	Value Type	Restrictions	Semantics
Arrival	Float min, max for each edge		The external insertion delay leading up to the clock root

Common Insertion Delay

Specifies the portion of the external insertion delay (see Clock Arrival) that is common to two clock roots

Type Environment condition

Applies To Pairs of clock roots (hierarchical pins or primitive outputs)

Depends On Clock, Derived Clock, Clock Arrival

Verified By None

Parameters

Name	Value Type	Restrictions	Semantics
Insertion delay	Float min, max for each edge	Non-negative	The external insertion delay shared by the two clock roots
Clock root 1	Pin name	Must be a clock root	The first clock root in the pair
Clock root 2	Pin name	Must be a clock root	The second clock root in the pair

Clock Mode

Specifies the default analysis for clock network delays

Type Control

Applies To None

Depends On None

Verified By None

Parameters

Name	Value Type	Restrictions	Semantics
Analysis mode	ideal or actual		This specifies whether the ideal insertion delay, skew, and transition times within all clock networks should be used, or whether the actual values should be computed. Generally the analysis mode is set to ideal prior to inserting the clock networks, and actual afterwards. This can be overridden by the Clock Delay specification.

Tree Mode

Specifies the default analysis for general buffer tree delays

Type Control

Applies To None

Depends On None

Verified By None

Parameters

Name	Value Type	Restrictions	Semantics
Analysis mode	ideal or actual		This specifies whether the ideal insertion delay, skew, and transition times within all general buffer trees described by Tree Delay should be used, or whether the actual values should be computed. Generally the analysis mode is set to ideal prior to inserting the buffer trees, and actual afterwards. This can be overridden by the Tree Delay specification.

Clock Delay

Constraint on the timing characteristics of a clock network or a portion of a hierarchical clock network

Type Constraint

Applies To Hierarchical pins, primitive outputs, or input pins on primitives which contain an internal clock network with significant insertion delay

Depends On Clock

Verified By Timing analysis

Parameters

Name	Value Type	Restrictions	Semantics
Root	Pin name		Name of hierarchical port or primitive output pin which drives the clock network, or an input pin on a primitive which contains an internal clock network with significant insertion delay. Either a pin name or a waveform name may be specified as the root, but not both. Values specified using a pin name override those specified using a waveform name.
	Waveform name		When a waveform name is given, it specifies default values which affect all clock networks driven by that waveform, including virtual clock networks referenced in ***Arrival*** and ***Required Time*** constraints.
Analysis mode	ideal or actual		This specifies whether the ideal insertion delay, skew, and transition times within the clock network should be used, or whether the actual values should be computed. Generally the analysis mode is set to ideal prior to inserting the clock network, and actual afterwards. This overrides the default Clock Mode specification.
Explicit leaf pins	List of pin names		These override the default rules, specifying that pins which would otherwise lie within the default clock network should be treated as leaf pins, or grouping a set of default leaf pins in order to override the default insertion delay or slew.
Data leaf pins	List of pin names		These are a special case of explicit leaf pins, where the logic beyond the leaf pin should be treated as part of the data network instead of the clock network.

Default insertion delay	Float for each edge	Non-negative	The nominal cumulative delay from the root to the default leaf pins, as well as the default value for explicit leaf pins.
Explicit insertion delay	Float for each edge	Non-negative	This overrides the default, specifying a different insertion delay for a group of explicit leaf pins
Internal insertion delay	Float min, max for each edge	Non-negative	Additional insertion delay that lies beyond an explicit leaf pin but should be included in the calculated insertion delay to the leaf. This is usually used when the explicit leaf pin is an input on a model of a hierarchical module, to represent the internal insertion delay from the input port to the real leaf pins within the module.
Default skew	Float min, max for each edge	Non-negative	The range of differences in insertion delay allowed between any pair of leaf pins, except for the excluded leaf pins.
Default transition time	Float min, max for each edge	Non-negative	The range of transition (ramp) times allowed at each default leaf pin, as well as the default value for explicit leaf pins.
Explicit transition time	Float min, max for each edge	Non-negative	The range of transition (ramp) times allowed at a group of explicit leaf pins.
Default overshoot limit	Float min, max		For board level analysis, the default overshoot allowed at each default leaf pin, as well as the default value for explicit leaf pins
Explicit overshoot limit	Float min, max		For board level analysis, the overshoot allowed at a group of explicit leaf pins

Tree Delay

Constraint on the timing characteristics of a general buffer tree

Type Constraint

Applies To Hierarchical pins or primitive outputs

Depends On None

Verified By Timing analysis

Parameters

Name	Value Type	Restrictions	Semantics
Root	Pin name		Name of hierarchical port or primitive output pin which drives the buffer tree
Analysis mode	ideal or actual		This specifies whether the ideal insertion delay, skew, and transition times within the buffer tree should be used, or whether the actual values should be computed. Generally the analysis mode is set to ideal prior to inserting the buffer tree, and actual afterwards. This overrides the default Tree Mode specification.
Explicit leaf pins	List of pin names		These override the default rules, specifying that pins which would otherwise lie within the default buffer tree should be treated as leaf pins, or grouping a set of default leaf pins in order to override the default insertion delay or slew.
Internal insertion delay	Float min, max for each edge	Non-negative	Additional insertion delay that lies beyond an explicit leaf pin but should be included in the calculated insertion delay to the leaf. This is usually used when the explicit leaf pin is an input on a model of a hierarchical module, to represent the internal insertion delay from the input port to the real leaf pins within the module.
Default insertion delay	Float for each edge	Non-negative	The nominal cumulative delay from the root to the default leaf pins, as well as the default value for explicit leaf pins.
Explicit insertion delay	Float for each edge	Non-negative	This overrides the default, specifying a different insertion delay for a group of explicit leaf pins
Default skew	Float min, max for each edge	Non-negative	The range of differences in insertion delay allowed between any pair of leaf pins, except for the excluded leaf pins.
Default transition time	Float min, max for each edge	Non-negative	The range of transition (ramp) times allowed at each default leaf pin, as well as the default value for explicit leaf pins.

Explicit transition time	Float min, max for each edge	Non-negative	The range of transition (ramp) times allowed at a group of explicit leaf pins.

Inter-Clock Uncertainty

Worst-case uncertainty between two clock distribution networks for harmonically related clocks

Type Constraint

Applies To Clock pins

Depends On ***Clock, Waveform***

Verified By None

Parameters

Name	Value Type	Restrictions	Semantics
Source root	Waveform name		When a waveform name is given, it specifies default values that affect all source clock networks driven by that waveform, including virtual clock networks referenced in ***Arrival*** constraints.
Target root	Waveform name		When a waveform name is given, it specifies default values that affect all target clock networks driven by that waveform, including virtual clock networks referenced in ***Required Time*** constraints.
Source launch edge	Rise/fall		Edge of the clock which launches data from the source register of a path between the source and target clock networks.
Target capture edge	Rise/fall		Edge of the clock which captures data at the target register of a path between the source and target clock networks.
Calculation mode	Absolute or increment		When absolute is specified, it indicates that the uncertainty value represents the entire skew between the clocks. When increment is specified, the uncertainty value is added to any skew calculated between the clocks
Propagation mode	Ideal or actual		When ideal is specified, the uncertainty value applies when either the clock network for the source register or the clock network for the target register is analyzed in ideal mode. When actual is specified, the uncertainty value applies when the clock networks to both the source and target registers are analyzed in actual mode.

Uncertainty	Float, min and max	The uncertainty value. This uncertainty may override or add to any skew inherent in the harmonic relationship between the edges in the source and target clock waveforms, and any skew introduced by the insertion delays in the clock networks to the source and target registers.

Target-Based Clock Uncertainty
Worst-case uncertainty between the clock edge at a target register and the clock edges at any source register
Type Constraint
Applies To Target clock leaf pins (possibly implied by a waveform, clock root, or register)
Depends On ***Clock, Derived Clock, Waveform, Derived Waveform***
Verified By None
Parameters

Name	Value Type	Restrictions	Semantics
Target	Clock root (pin name)		Name of pin that is the root of a clock distribution network
	Clock leaf (pin or instance name)		Name of clock input pin on a target register, or the name of the target register instance
	Waveform name		When a waveform name is given, it specifies default values that affect all target registers that are leaves of clock networks driven by that waveform, including virtual clock networks referenced in ***Required Time*** constraints.
Calculation mode	Absolute or increment		When absolute is specified, it indicates that the uncertainty value represents the entire skew between the target clock and any source clock. When increment is specified, the uncertainty value is added to any skew calculated between the target clock and any source clock.
Propagation mode	Ideal or actual		When ideal is specified, the uncertainty value applies when the clock network for the target register is analyzed in ideal mode. When actual is specified, the uncertainty value applies when the clock network to the target register is analyzed in actual mode.
Uncertainty	Float min, max for each edge		The uncertainty value. This uncertainty may override or add to any skew inherent in the harmonic relationship between the edges in the source and target clock waveforms, and any skew introduced by the insertion delays in the clock networks to the source and target registers.

Derived Waveform

A waveform which is derived from another waveform (by a clock multiplier or divider, for example)

Type Environment condition

Applies To None

Depends On None

Verified By None

Parameters

Name	Value Type	Restrictions	Semantics
Parent	String		The name of the waveform from which this waveform is derived
Inverted	Boolean		Whether the derived waveform is inverted or not
Phase shift	Float for each edge		Offset of each edge from the parent waveform
Frequency multiplier	Integer	Positive	Multiplier relative to parent waveform frequency. Both the frequency multiplier and divisor can be specified to indicate a rational ratio between the parent and derived frequencies.
Frequency divisor	Integer	Positive	Divisor relative to parent waveform frequency
Jitter adjustment	Float min, max per edge	Non-negative	Potential positive, negative variation in actual edge position relative to the ideal edge position across different clock cycles.

Waveform

A description of an abstract waveform

Type Environment condition

Applies To None

Depends On None

Verified By None

Parameters

Name	Value Type	Restrictions	Semantics
Ideal edges	List of floats		Offsets from an implicit reference point in time
Jitter	Float min, max per edge	Non-negative	Potential positive, negative variation in actual edge position relative to the ideal edge position across different clock cycles.
Domain	String		Name of a clock domain. All clocks in the same domain are synchronous with respect to each other.

1.2 Timing Exceptions

Multicycle Path (set_cycle_addition)

Paths are described using from/through/to parameters and the specified cycle_value is added to the paths. This can be used to describe multicycle paths, 1/2 cycle paths, etc.

Type Constraint

Applies To Paths

Depends On None

Verified By Timing analysis

Parameters

Name	Value Type	Restrictions	Semantics
From	List of pins		(see To parameter semantics)
Through	List of pins		(see To parameter semantics)
To	List of pins		From/Through/To are used to describe paths that span multiple cycles. Note there can be multiple throughs. Each parameter can be a list. A path matches the pattern if at least one pin in the parameter list is on the path, and each parameter is satisfied. The order of the throughs is also important.
from_clock	clock_name		(See multicycle path to_clock parameter)
to_clock	clock_name		This is similar to the from/to parameter, except that all registers triggered by the specified clock are considered as the from/to of the path.
from_edge	rise/fall		The edge on the from pin(s).
through_edge	rise/fall		The edge on the through pin(s).
to_edge	rise/fall		The edge on the to pin(s).
analysis_type	early/late		The early paths can be described separately from the late paths.
cycle_value	float		The number of cycles that are added to the paths described by the from/through/to parameters. Note, all paths by default have one cycle. The cycle_value for a two cycle paths would be 1. Negative numbers and fractions can also be used to get 1/2 cycle paths.

Disable Path (set_false_path)

Paths are described using from/through/to parameters and these paths are disabled.

Type Constraint

Applies To Paths

Depends On None

Verified By Timing Analysis

Parameters

Name	Value Type	Restrictions	Semantics
From	List of pins		(see To parameter semantics)
Through	List of pins		(see To parameter semantics)
To	List of pins		From/Through/To are used to describe paths that should be disabled. Note there can be multiple throughs. Each parameter can be a list. A path matches the pattern if at least one pin in the parameter list is on the path, and each parameter is satisfied. The order of the throughs is also important.
from_clock	clock_name		(See multicycle path to_clock parameter)
to_clock	clock_name		This is similar to the from/to parameter, except that all registers triggered by the specified clock are considered as the from/to of the path.
from_edge	rise/fall		The edge on the from pin(s).
through_edge	rise/fall		The edge on the through pin(s).
to_edge	rise/fall		The edge on the to pin(s).
analysis_type	early/late		The early paths can be described separately from the late paths.

Path Delay

The maximum (or minimum) delay is described using from/thru/to parameters.

Type Constraint

Applies To Paths

Depends On None

Verified By Timing Analysis

Parameters

Name	Value Type	Restrictions	Semantics
From	List of pins		(see To parameter semantics)

Through	List of pins		(see To parameter semantics)
To	List of pins		From/Through/To are used to describe paths that have a maximum (or minimum) delay. Note there can be multiple throughs. Each parameter can be a list. A path matches the pattern if at least one pin in the parameter list is on the path, and each parameter is satisfied. The order of the throughs is also important.
from_clock	clock_name		(See multicycle path to_clock parameter)
to_clock	clock_name		This is similar to the from/to parameter, except that all registers triggered by the specified clock are considered as the from/to of the path.
from_edge	rise/fall		The edge on the from pin(s).
through_edge	rise/fall		The edge on the through pin(s).
to_edge	rise/fall		The edge on the to pin(s).
analysis_type	early/late		The early (min) paths can be described separately from the late (max) paths.

Disable Arc (set_disable_cell_timing)

Disables a timing arc on a library cell, disabling the timing arc on all instances of that cell

Type Constraint

Applies To Library Cell

Depends On None

Verified By Timing Analysis

Parameters

Name	Value Type	Restrictions	Semantics
library_name	Library name		The library where the cell is found.
cell_name	Cell name		The cell on which the arc is to be disabled.
from_pin	Pin name		The from pin on the arc.
to_pin	Pin name		The to pin on the arc.

1.3 Timing Boundary Conditions

Arrival Time

Specifies a partial path delay-time-range arriving at an input, a bi-directional port instance, or an internal pin, not including interconnect and loading delays due to the net external to the port, thus provides the constraint for the timing allowed for the remaining path internal to the block. If placed on an internal pin, it will over-ride any propagated values.

Type Constraint

Applies To Hierarchical or primitive input or bi-directional pins

Depends On Clock domain

Verified By Timing Verifier

Parameters

Name	Value Type	Restrictions	Semantics
Port Instance or Internal Pin	Port identifier or Internal Pin	Input or bi-directional port or Internal Pin	Name of the port or internal pin.
Time Reference Point	Waveform-edge identifier	Waveform must be previously defined	The point of reference for the timing calculation. If omitted, the implicit reference point of time is assumed.
Rise-time-range	Float, min-max offset from the waveform-edge	At least, one of the two must be specified. In multiple specifications for the same pin relative to the same waveform edge, only the latest one is in effect.	Earliest-latest delay of a rising transition (Note: negative values allowed.)
Fall-time-range	Float, min-max offset from the waveform-edge		Earliest-latest delay of a falling transition (Note: negative values allowed.)

Departure Time

Specifies a partial path delay-time-range beyond an output, internal pin, or a bi-directional port instance, not including interconnect and loading delays due to the net external to the port, thus provides the constraint for the timing reserved for the remaining path external to the block. If placed on an internal pin, it will over-ride any propagated values.

Type Constraint

Applies To Hierarchical or primitive output or bi-directional pins

Depends On Clock domain

Verified By Timing Verifier

Parameters

Name	Value Type	Restrictions	Semantics
Port Instance or Internal Pin	Port identifier or Internal Pin	output or bi-directional port or internal pin	Name of the port or internal pin
Time Reference Point	Waveform-edge identifier	Waveform must be previously defined	The point of reference for the timing calculation. If omitted, the implicit reference point of time is assumed.
Setup Rise-time-range	Float, min-max offset from the waveform-edge	At least, one of the two must be specified. In multiple specifications for the same pin relative to the same waveform edge, only the latest one is in effect.	Earliest-latest delay of a rising transition evaluated with respect to the setup condition of the target register. (Note: negative values allowed.)
Setup Fall-time-range	Float, min-max offset from the waveform-edge		Earliest-latest delay of a falling transition evaluated with respect to the setup condition of the target register. (Note: negative values allowed.)
Hold Rise-time-range	Float, min-max offset from the waveform-edge	At least, one of the two must be specified. In multiple specifications for the same pin relative to the same waveform edge, only the latest one is in effect.	Earliest-latest delay of a rising transition evaluated with respect to the hold condition of the target register. (Note: negative values allowed.)

Hold Fall-time-range	Float, min-max offset from the waveform-edge	Earliest-latest delay of a falling transition evaluated with respect to the hold condition of the target register. (Note: negative values allowed.)

Required Time

Specifies the required time at the specified output, bidir, or internal pin from a specified time reference point. (This is different from the Departure Time, which is the delay to a specified time reference point.)

Type Constraint

Applies To Internal pins, output or bidir ports

Depends On None

Verified By Timing Verifier

Parameters

Name	Value Type	Restrictions	Semantics
Port Instance or Internal Pins	Port identifier or Internal Pin	output or bi-directional port or internal pin	Name of the port or internal pin
Time Reference Point	Waveform-edge identifier	Waveform must be previously defined	The point of reference for the timing calculation. If omitted, the implicit reference point of time is assumed.
Rise-time-range	Float, min-max offset from the waveform-edge	Both range must be specified. In multiple specifications for the same pin relative to the same waveform edge, only the latest one is in effect.	Earliest-latest delay of a rising transition (Note: negative values allowed.)
Fall-time-range	Float, min-max offset from the waveform-edge		Earliest-latest delay of a falling transition (Note: negative values allowed.)

External Delay

Specifies the min-max delay range of a combinatory partial path external to an I/O port, excluding the interconnect and loading delays of the connecting net. These delays are added to the delay inside the design, then compared to the Path Delay.

Type Constraint

Applies To I/O ports, Internal pins, or primitive inputs/outputs

Depends On None

Verified By Timing Verifier

Parameters

Name	Value Type	Restrictions	Semantics
Port Instance or Internal Pin	Port identifier or internal pin	output or bi-directional port	Name of the port
Rise-time-range	Float, min-max offset from the waveform-edge	Both range must be specified. In multiple specifications for the same pin relative to the same waveform edge, only the latest one is in effect.	Earliest-latest delay of a rising transition (Note: negative values allowed.)
Fall-time-range	Float, min-max offset from the waveform-edge		Earliest-latest delay of a falling transition (Note: negative values allowed.)

Driver Specification

Specifies the characteristics for the driver that is driving the external net connecting to an input or a bi-directional port

Type Environment condition

Applies To Primitive input or bi-directional pins

Depends On None

Verified By None

Parameters

Name	Value Type	Restrictions	Semantics
Port Instance	Port identifier	output or bi-directional port	Name of the port
Driver Cell	Cell Output Port identifier	Cell must included in the cell library	Define the cell output to be used to calculate the loading delay for the external net
Driver Strength	Float	Strength value must be consistent with those specified in the cell library	Alternatively, define the cell output drive strength to be used to calculate the loading delay for the external net
Input Slew	Float		Assigning a input slew to the driver cell for the purpose of calculating the loading delay of the external net.

1.4 Parasitic Boundary Conditions

<<<
****
*

DCWG reviewers: This draft section is best described as being in 3 parts.

- A general discussion of parasitics from the point of view of external or interface ports of a block definition and instance (potentially occurrence) ports of a block instance (boundary conditions). I hope the terminology I use wont get in the way of your review, however, once one goes fully hierarchical, then we have to become more specific than, for instance, ‘external’. The discussion also covers the several levels of parasitics that are coming, e.g., distributed RC, RCL, and RCLM, all which need to include coupling.
- My current list of list of hierarchical parasitics at the boundaries of blocks in a design hierarchy. I am in the process of redoing the appropriate ones to reflect those offer within the Ambit synthesis constraints document.
- A first cut at fleshing the list of hierarchical parasitics at the boundaries of blocks in a design hierarchy

****
>>>

This section defines the Parasitic Hierarchical Boundary Conditions needed to define and constrain the timing of signals being carried by a net and applicable operating environment conditions. These may be defined from within the design descriptions and/or from tools and/or by the designer/user.

The scope of this section includes the following coverage and points of view:

- This section defines the boundary conditions that surround any particular net or fragment of a set within a block within a design hierarchy. The boundary conditions therefore define the specific environment for defining (prescribing) and/or constraining the timing of signals being carried by a net.
- Electrical, magnetic, optical, thermal, (and possibly other forms of energy), material, and mechanical properties that drive (source, transmit), load (sink, receive, terminate), surround, or otherwise affect the timing of signals carried by a net.
- The timing of signals on the net may be affected directly or indirectly by the above forms of energy, mechanics, and material properties. These parasitics may therefore include effects of coupling and ‘over the cell’ routing.
- Presumed, typical, or actual characteristics of the signals being carried (propagated) to the net through some port or via. Both static, steady state and transient signal characteristics may be prescribed.
- Presumed, typical, or actual characteristics of the environment conditions, including, but not limited to, power, temperature, particle density, noise. Both static, steady state and transient characteristics may be prescribed.
- Hierarchical structure of that portion of the design surrounding the net:
- Target net is assumed to be contained within one block of an overall design hierarchy. The target net may also possibly be further contained within one layer or strata of the physically implemented design. Where a net consists of more than one net fragment (portion, net segment, subnet), the fragments may be connected by a via or other intra-layer conducting mechanism.
- Hierarchical driving boundary conditions: The signals carried by a net may pass from an upper level portion of the design hierarchy to the target net within a block by flowing from a receiving instance port (occurrence port in the fully elaborated hierarchical design) ‘down’ to the containing block’s interface definition port. Signals may also be carried from a lower level portion (sub-block or block instance) of the design hierarchy to the net by flowing ‘up’ from a sub-block’s driving instance port.
- Hierarchical receiving boundary conditions: The signals are carried away from a net

to some upper level portion of the design hierarchy from the target net within a block by flowing to the containing block's receiving interface definition port 'up' to the above portion of the design hierarchy. Signals may also be carried away from a net to some lower level portion of the design hierarchy by flowing 'down' to a sub-block's receiving instance port.

- Hierarchical Boundary Parasitics Levels: are defined for several levels of detail (aka levels of abstraction), including the following:
- High-level functional design parasitics boundary conditions
- Logic design parasitic boundary conditions
- constraints
- Lumped parasitics - generally applicable to 350nm and above. Coupling is via lumped capacitance. (This is the level of detail currently in wide spread use.)
- Distributed R-C parasitics - generally applicable t 250nm to 180nm. Coupling is via distributed capacitance. (This is the level of detail just coming into wide spread practice.)
- Distributed R-L-C parasitics - generally applicable to 180nm to perhaps around 130nm. Coupling is via distributed capacitance, with mutual coupling considered being negligible.
- Distributed R-L-C-M parasitics - generally applicable to 130nm to 100nm and below. Coupling is via distributed capacitance and distributed mutual inductance.

Hierarchical Boundary Parasitics

- In CHDS and CHDStd, we are moving to separate the representation of interconnect (and its parasitics) associated with the interface circuitry of the block from the representation of interconnect (and its parasitics) associated with the internal definition of the block. The paradigm of separating interfacing effects from internal effects is well known. The use of hierarchy, complex interface and internal interconnect parasitics, and the growing amount of coupling between nets are part of the approach needed to handlethe growing complexity at 180nm and below.
- Interface interconnect is always instance and occurrence specific. The parasitics of the interconnect and its surrounding drivers and loads that are reflected at the interface ports are likewise instance and occurrence specific.
- Internal interconnect is always block_definition specific. The parasitics of instance ports within the block's internal definition, once resolved with the instantiated lower-level circuitry, are normally constant with each block instance.
- External coupling or over-the-cell route coupling with internal interconnect requires treating the coupling-affected portion as part of the interface interconnect. This is because the external coupling is by definition a form of interface access into the block definition.
- An interface port of a block (cell, macro, core) definition may normally be treated as two components: an open-circuit output (or input) impedance, and separately, the loaded incremental impedance,
- Open-Circuit (Unterminated) Impedance - Such open circuit impedance may be represented by a R-C, R-L-C, or R-L-C-M parasitic model of the interconnect interface circuitry "behind" the interface port , including the effects of nets, vias, and net-to-net coupling. The parasitic model of the unloaded interface port is associated with the port
- Loaded (Terminated) Incremental Impedance – the loaded incremental impedance may also be represented by a R-C, R-L-C, or R-L-C-M parasitic model of the interconnect instance port interface circuitry "looking" into the interface port of the instantiated block.
- This should not include the parasitic model effects of the ports the drive or receive from the target port. Those effects are accounted for at those other ports.

The above effects and requirements need to be reflected in the below list of hierarchical parasitic boundary conditions:

The following are the proposed list of terms for hierarchical parasitic boundary conditions:

1.4.1 Functional Design Boundary Conditions

1.4.1.1 Fanout

1.4.1.1.1 Fanout Drive

1.4.1.1.2 Fanout Load (Ambit)

Ambit Command: `set_fanout_load`

Syntax: set_fanout_load <load> <port_list>

Ambit Description: The `set_fanout_load` command is used to specify the fanout load on the ports of a cell. While port capacitance affects timing analysis, fanout loads are used to enforce the design rule checks.

DCWG Description: `fanout_load:` Specifies the expected or budgeted fanout load on the ports of a cell when the cell is instantiated. While port capacitance affects timing analysis, fanout loads are used during planning the loading of a cell, and later to enforce the design rule checks of the application or use a cell.

Arguments:

Ambit Arguments:

Load; Total fanout load external to the cell.

`port_list` — List of ports.

DCWG Load; Total fanout load external to the cell.

`port_list` — List of ports.

Options: `None`

Attributes: fanout_load_limit

Design Database:

Related Commands: set_current_module

1.4.1.1.3 Fanout Load Limit (Ambit)

1.4.1.1.3.1 set_fanout_load_limit

Command: `set_fanout_load_limit`

Syntax: `set_fanout_load_limit <load port_list>`

Description: The set_fanout_load_limit command is used to specify the fanout load limit (maximum value) on the ports of a cell. While port capacitance affects timing analysis, fanout loads are used to enforce the design rule checks. The design rule requirement of a maximum fanout load value is set using the global attribute fanout_load_limit.

This command overrides the default fanout load limit on specific ports set by the global attribute `fanout_load_limit`.

Arguments: `load` — Fanout load limit on the ports.

`port_list` — List of ports.

Options: None

Attributes: fanout_load_limit

Design Database:

Related Commands: set_current_module, set_fanout_load

Examples: `set_fanout_load_limit 4 [find -port -output data*]`

1.4.1.1.3 External Fanout

Identifies the number of effective loads (transistor circuits) lying behind an output port.

Type

Applies To

Depends On

Verified By

Parameters

Name	Value Type	Restrictions	Semantics

1.4.1.2 Equivalent External Sources (Ambit – num_external_sources)

1.4.1.2.1 set_num_external_sources

Command: `set_num_external_sources`

Syntax: set_num_external_sources <num_sources> <port_list>

Description: The set_num_external_sources command can be specified on top level input and output ports. This command sets the constraint that the number of external sources specified by the command are connected to the ports in the port list. This number is factored into the wire capacitance and wire resistance estimation done for the port nets using the wire load models.

Arguments: num_sources — Number of external sources
port_list — List of ports

Options:

Attributes:

Design Database:

Related Commands:

Examples: `set_num_external_sinks`
`set_port_capacitance`
`do_derive_context`
`do_time_budget`

1.4.1.3 Fanin

1.4.1.3.1 Equivalent Fanin Loads

1.4.1.3.2 Equivalent External Sinks (Ambit – num_external_sinks)

1.4.1.3.2.1 set_num_external_sinks

Command: `set_num_external_sinks`

Syntax: `set_num_external_sinks <num_sinks> <port_list>`

Description: The set_num_external_sinks command can be specified on top level input and output ports. This command sets the constraint that the number of external sinks specified by the command are connected to the ports in the port list. This number is factored into the wire capacitance and the wire resistance estimation done for the port nets using the wire load models. It does not add towards the total fanout count for Design Rule Violation (DRV).

If you do not annotate set_num_external_sinks, then the fanout is 1 for the port (and any other internal sinks of the wire). If you specify set_num_external_sinks, then you get that exact number added to the count of internal sinks of the wire. Therefore, setting set_num_external_sinks 1 has no effect.

For more information on design rule violations, see the description for "set_fanout_load".

Arguments: `num_sinks` — Number of external sinks

`port_list` — List of ports

Options:

Attributes

Design Database:

Related Commands: set_num_external_sources, set_port_capacitance, do_derive_context, do_time_budget, set_port_wire_load, set_fanout_load

Examples:

1.4.1.4 External load

An example portion of External Load would be the lumped capacitance that is or could be loading an output port or pin.

Type

Applies To

Depends On

Verified By

Parameters

Name	Value Type	Restrictions	Semantics

1.4.2 *Logic Design Boundary Conditions*

1.4.2.1 *Drive Capability*

1.4.2.1.1 Cell (Block) Drive (Ambit)

1.4.2.1.2 Drive resistance (Ambit)

1.4.2.1.2.1 set_drive_resistance

Command: `set_drive_resistance`

Syntax: `set_drive_resistance ?-rise | -fall? ?-early | -late? value port_list`

Description: The set_drive_resistance command is a simpler version of the set_drive_cell command and can be used in many situations where the drive resistance can be specified. It is only used for timing analysis. It does not affect the electrical properties of the design. It is used to specify the drive resistance of a cell. It computes an offset to the arrival time of an input and also changes the slew time used to compute the delay of the cell on the sink of the net.

The arrival time at the input port is modified by adding the RC constant to the specified arrival time at the input port. The RC constant is the capacitance(C) seen at the input port multiplied by the drive resistance (R). The RC value is used as the slew value for the delay calculation of the next cell. This command overrides the set_drive_cell command per port if it is the last command applied. Also, if set_drive_resistance is set for a port, set_slew_time is ignored for that port.

Arguments: `value` — Resistance value.

`port_list` — List of ports for which drive resistance is specified.

Options: `-rise` – Specifies that the drive resistance is applicable to only the rising edge transition at the input port.

`-fall` – Specifies that the resistance is applicable only to the falling edge transition at the input port. If either `rise` nor `fall` options are specified then the

resistance is applied to both transitions at the input port.
`-early` – Specifies that the drive resistance should be applied to the early arrival time (hold time) for timing analysis.
`-late` – Specifies that the drive resistance should be applied to the late arrival time (setup time) for timing analysis.
Attributes
Design Database:
Related Commands: set_drive_cell
set_slew_time
Examples: `set_drive_resistance`
```
[expr 3 * [get_cell_drive -cell IV –pin A] ]
[-input -port * ]
```

1.4.2.2 Available Drive

1.4.2.3 Drive Strength –cell(s) input ports.

Semantic definition:

1.4.2.3.1 Input ports:

Identifies the amount of transistor circuit output capability available to drive in to an input port from some higher or lower level circuitry.

1.4.2.3.2 Output ports:

Identifies the amount of transistor circuit output capability available to drive out through an output port to some higher or lower level circuitry.

Type

Applies To

Depends On

Verified By

Parameters

Name	Value Type	Restrictions	Semantics

1.4.2.4 Equivalent Load

1.4.3 Lumped Parasitic Boundary Conditions

1.4.3.1 Effective Load Capacity

1.4.3.2 Port Capacitance

1.4.3.2.1 set_port_capacitance (Ambit)

Command: `set_port_capacitance`

Syntax:: set_port_capacitance <capacitance> <port_list>

Description: The `set_port_capacitance` command specifies the capacitance external to the design based on input and output loading from other ports and nets connected to the ports of the current module.

The capacitance at any port is the sum of the external port capacitances the port is connected to. For an input port the port capacitance refers to the capacitance of all the ports of the driver that are connected to the net and the capacitance of the other loads on the net. For output ports, the port capacitance refers to the capacitance of all the external sinks, or external drivers, for the case of multiply-driven net, that are connected to the net as well as the capacitance of the input port of any other external drivers

Arguments:

`capacitance` — Capacitance value.

port_list — List of ports.

Options: None

Attributes: capacitance_limit

Design Database: None

Related Commands: set_current_module, set_port_capacitance_limit

Examples: This command sets a 3.2 capacitance on all output ports of the current module whose names match dbus at the start The unit of capacitance will be the same as the unit of capacitance used in the library.

```
set_port_capacitance 3.2 [find -port –
output bus*]
```

1.4.3.2.2 set_port_capacitance_limit (Ambit)

Command: set_port_capacitance_limit

Syntax: set_port_capacitance_limit <capacitance> <port_list>

Description: The set_port_capacitance_limit command specifies the limit on the capacitance (maximum value) external to the design based on input and output loading from other ports and nets connected to the ports of the current module. It is used to override design rule constraints set by the global attribute **set_capacitance_limit:** These constraints can be specified on top level input and output ports. It sets the constraint that the total capacitances (i.e wire capacitance nd pin capacitance) of nets attached to the ports in the port list do not exceed the specified capacitance limit. This command overrides the default limit set by the `set_global_capacitance_limit` command.

Arguments:

`capacitance` — Capacitance value.

`port_list` — List of ports.

Options: None

Attributes: capacitance_limit

Design Database:

Related Commands: set_fanout_load_limit, set_slew_limit, set_port_capacitance, set_num_external_sources, set_num_external_sinks, do_derive_context, do_time_budget, set_global capacitance_limit

Examples: set_port_capacitance_limit 5.0 [find -port dbus*]

1.4.3.2.2.1 Input Port Capacitance

1.4.3.2.2.2 Input Port Capacitance Limit

1.4.3.2.2.3 Output Port Capacitance

1.4.3.2.2.4 Output Port Capacitance Limit

1.4.3.3 Effective Load Resistance

1.4.3.4 Effective Drive Capacitance

1.4.3.5 Effective Drive Resistance

1.4.3.6 Equivalent Custom Wire Load

While wire load models normally apply to the target net, during that portion of the design flow where wire load parasitics are being used, we will need also to encapsulate and store the effective amount of wireload model that is present at the hierarchical port or port instance.

1.4.3.6.1 Equivalent Custom Wire Load Mode (Ambit)

1.4.3.6.2 Specified Wire Load Model (Ambit)

1.4.3.7 Port Wire Load (Ambit)

1.4.3.7.1 set_port_wire_load

Command: set_port_wire_load

Syntax: set_port_wire_load ?-library <library_name>? <wire_load> `<port_list>`

Description: The set_port_wire_load command specifies the wire load for either an input or output top level port of a design. The net connected to the port is associated with the specified wire load model, which is used for wire cap and resistance estimation. If a net is connected to more than one port with a set_port_wire_load assertion, then the worst wire load model is computed and used for the net. The library name indicates the location of the specified wire load model. If the library is not specified, then the wire load is located from the target technology (the default location).

Arguments:

Options:

Attributes:

Design Database:

Related Commands: set_wire_load, set_num_external_sinks

Examples:

1.4.3.7.2 Input Port Wire Load (Ambit)

See set_port_wire_load.

1.4.3.7.3 Output Port Wire Load (Ambit)

See set_port_wire_load.

1.4.3.7.4 Equivalent Custom Wire Drive - ???

1.4.3.8 Lumped RC parasitic model

Semantic definition: The distributed parasitic effects of a net or netsegment are represented by a single lumped value of capacitance and resistance. Nominal, best-case, and worst-case RC value pairs may be specificed from which to calculate nominal, best-case, and/or worst-case net or netsegment delay, slew, and other related timing characteristics.

Type: Parasitic

Applies To: Hierarchical pins or primitive outputs

Depends On: process: Coefficient

Verified By:

Parameters

Name	Value Type	Restrictions	Semantics

1.4.4 Distributed Hierarchical Boundary Parasitics (Impedance)

This section defines the distributed R-C, R-L-C, and R-L-C-M parasitic boundary conditions for initial estimation of instantiating blocks, cells, and macros, and for later backannotated actual parasitics.

The approach described in this section currently is being examined for applicability to fully accurate hierarchical net timing stored within the folded (instance) design hierarchy. (I recently found published information that this approach is in used within a major EDA supplier.)

The analysis and work todate indicates that at least of large number of nets may be accurately calculated from the below hierarchical two port parasitic boundary conditions if the timing is then captured arc-wise in a timing view which also have both internal and interface timing arcs. An additional instance-port incremental timing offset value is required at the ‘edge’ of the block-instance’s interface timing view as part of representing the information required for an full instantiation of a hierarchical net parasitic model.

1.4.4.1 Unterminated (Open Circuit) Interface Port Parasitic (impedance) model

The Unterminated interface port parasitics model encapsulates the effects of the interface nets behind the port.

1.4.4.2 Terminated (Loaded) Instance (Occurrence) Port Parasitic (impedance) model

The terminated instance port parasitics model encapsulates the incremental effects of the interface nets as (potentially) loaded by the parasitics of the nets at the next level of the design hierarchy.

1.4.4.3 Distributed R-C Port Parasitics

1.4.4.4 Distributed R-C-L Port Parasitics

1.4.4.5 Distributed R-C-L-M Port Parasitics

1.5 Operating Conditions