7 Timing and Power Analyses

Other than circuit simulation that gives output waveform of circuit behavior, the HSIM simulator also provides a set of timing and power checking commands to perform timing and power analysis. Each command needs to be specified within the netlist file, or in a separate file which is included (using **.include** option) into the netlist file. The timing check command starts with the keyword **.tcheck** and is described in the following section. The power check command starts with the keywod **.pcheck** and is described in another section. If there is any timing/power check error, it will be reported in the file *hsim.chk*, or *out_file.chk* if "**-o** out_file" is specified when invoking the HSIM simulator.

7.1 Timing Checks

7.1.1 Setup Time Check

The setup time check command syntax is

> **.tcheck** *title_name* **setup** *sig_name edge_type ref_name ref_edge_type setup_time* <**subckt**=sub_name> <**window**=*window_limit*> <**vlth**=*logic_low_voltage*> <**vhth**=*logic_high_voltage*> <**refvlth**=*ref_logic_low_voltage*> <**refvhth**=*ref_logic_high_voltage*>

The *title_name* defines the title name of this setup time check. Any setup time error resulted from this timing check will be listed in the timing/power check error file, and starts with this title name. The name specified by *sig_name* defines signal node name(s), which can be the node name of a single node or a node name containing wildcard character '*' representing a group of node names. Since the wildcard character '*' is also treated as a comment character in SPICE syntax such that all characters after '*' are treated as comment and ignored by HSIM netlist parser, the node name with wildcard character is thus required to be specified as v(node_name) or 'node_name'. The name *ref_name* defines the reference node name. *Edge_type* and *ref_edge_type*, which could be one of 'R'. 'F', and 'X', define the permissible state transition type for signal node and reference node, respectively: 'R' indicates it performs timing check only when the state transition is from '0' to '1'; 'F' indicates '1' to '0' transition; and 'X' indicates any state transition either from '0' to '1' or from '1' to '0'. The time specified by *setup_time* defines the setup time such that a setup timing error occurs when the time difference, between the permissible state transition time of reference node and the permissible state transition time of signal node, is less than *setup_time*.

If the **subckt** parameter is specified, the operation is performed to all instances of the particular subcircuit and the wildcard character '*' cannot be used. In addition, the *sig_name* and *ref_name* need to be the node names inside the subcircuit, not the full-path names.

When *setup_time* is positive and *window_limit* is not specified, the setup timing check error occurs when the signal node has a permissible transition at the time interval (*t_ref* - *setup_time*, *t_ref*). When *window_limit* is specified, the setup timing check error occurs when the signal node has a permissible transition at the time interval (*t_ref* - *setup_time*, *t_ref* + *window_limit*). In case *setup_time* is negative, it is required to specify *window_limit*.

The values *logic_low_voltage* and *logic_high_voltage* define the threshold voltage of logic '0' state and logic '1' state of signal node respectively. The signal node has a logic '0' state if its node voltage is lower than *logic_low_voltage*; it has a logic '1' state if the signal node voltage is higher than *logic_high_voltage*. The values *ref_logic_low_voltage* and *ref_logic_high_voltage* define the logic threshold voltages for the reference node. When *logic_low_voltage* is specified, *ref_logic_low_voltage* will be defined to the same value if it is not separately defined by **refvlth**. Similarly, when *logic_high_voltage* is specified, *ref_logic_high_voltage* will follow the same value if it is not separately defined by **refvhth**.

Examples:

> .**tcheck** check1 **setup** data1 x clk f 2.1n
> .**tcheck** check2 **setup** v(x1.*.data1) x clk f 2.1n
> .**tcheck** check33 **setup** d x clk r 1.5n **subckt**=dff

A setup time error occurs when any state transition at node *data1* occurs less than 2.1 ns before the fall transition of node *clk*. The error is reported following the title name of *check1*. Similar check will be performed for all nodes that match the pattern x1.*.data1 and the setup timing error, if any, is reported in the timing/power check file following the title name of *check2*. In the third example, the setup time check will report error for every dff subcircuit instance if the state transition at data pin d occurs less than 1.5 ns before the rise transition at the clk pin within the same dff subcircuit instance.

Example:

> .**tcheck** check3 **setup** qn f ck r -0.5n **window**=10n

A setup time error, following the title name of *check3,* is reported when a fall state transition at node *qn* occurs 0.5 ns after the node *ck* has a rise transition, and this fall state transition at *qn* occurs no later than 10 ns after the rise transition at node *ck*.

7.1.2 Hold Time Check

The hold time check command syntax is

> **.tcheck** *title_name* **hold** *sig_name edge_type ref_name ref_edge_type hold_time*
> <**subckt**=sub_name> <**window**=*window_limit*> <**vlth**=*logic_low_voltage*>
> <**vhth**=*logic_high_voltage*> <**refvlth**=*ref_logic_low_voltage*>
> <**refvhth**=*ref_logic_high_voltage*>

The *title_name* defines the title name of this hold time check. Any hold time error resulted from this timing check will be listed in the timing/power check error file, and starts with this title name. The name specified by *sig_name* defines signal node name(s), which can be the node name of a single node or a node name containing wildcard character ‘*’ representing a group of node names. Since the wildcard character ‘*’ is also treated as a comment character in SPICE syntax such that all characters after ‘*’ are treated as comment and ignored by HSIM netlist parser, the node name with wildcard character is thus required to be specified as v(node_name) or ‘node_name’. The name *ref_name* defines the reference node name. *Edge_type* and *ref_edge_type*, which could be one of ‘R’. ‘F’, and ‘X’, define the permissible state transition type for signal node and reference node, respectively: ‘R’ indicates it performs timing check only when the state transition is from ‘0’ to ‘1’; ‘F’ indicates ‘1’ to ‘0’ transition; and ‘X’ indicates any state transition either from ‘0’ to ‘1’ or from ‘1’ to ‘0’. The time specified by *hold_time* defines the hold time such that a hold timing error occurs when the time difference, between the permissible state transition time of signal node and the permissible state transition time of reference node, is less than *hold_time*.

If the **subckt** parameter is specified, the operation is performed to all instances of the particular subcircuit and the wildcard character ‘*’ cannot be used. In addition, the *sig_name* and *ref_name* need to be the node names inside the subcircuit, not the full-path names.

When *hold_time* is positive and *window_limit* is not specified, the hold timing check error occurs when the signal node has a permissible transition at the time interval (*t_ref*, *t_ref* + *hold_time*). When *window_limit* is specified, the hold timing check error occurs when the signal node has a permissible transition at the time interval (*t_ref* – *window_limit*, *t_ref* + *hold_time*). In case *hold_time* is negative, it is required to specify *window_limit*.

The values *logic_low_voltage* and *logic_high_voltage* define the threshold voltage of logic ‘0’ state and logic ‘1’ state of signal node respectively. The signal node has a logic ‘0’ state if its node voltage is lower than *logic_low_voltage*; it has a logic ‘1’ state if the signal node voltage is higher than *logic_high_voltage*. The values *ref_logic_low_voltage* and *ref_logic_high_voltage* define the logic threshold voltages for the reference node. When *logic_low_voltage* is specified, *ref_logic_low_voltage* will be defined to the same value if it is not separately defined by **refvlth**. Similarly, when *logic_high_voltage* is specified, *ref_logic_high_voltage* will follow the same value if it is not separately defined by **refvhth**.

Example:

.tcheck check3 **hold** data x clk f 2.1n

A hold time error occurs when any state transition at node *data* occurs at the time interval (t, t+2.1ns), where t is the time when node *clk* has a fall state transition The error is reported following the title name of *check3*.

Example:

.tcheck check4 **hold** qn f ck r -0.5n **window**=10n

A hold time error, following the title name of *check4*, is reported when a fall state transition at node *qn* occurs at the time interval (t – 10 ns, t – 0.5 ns), where t is the time when the node *ck* has a rise transition.

7.1.3 Pulse Width Check

This command checks whether the pulse width (time difference between rise and fall state transitions) of the specified node meets the required range. Its syntax is

> **.tcheck** *title_name* **pulsew** *node_name low_min_time low_max_time high_min_time high_max_time* <**subckt**=sub_name> <**vlth**=*logic_low_voltage*>
> <**vhth**=*logic_high_voltage*>

The *title_name* defines the title name of this pulse width check. Any violation against the required pulse width range resulted from this check will be listed in the timing/power check error file, and starts with this title name. The name specified by *node_name* defines signal node name(s), which can be the node name of a single node or a node name containing wildcard character '*' representing a group of node names. Since the wildcard character '*' is also treated as a comment character in SPICE syntax such that all characters after '*' are treated as comment and ignored by HSIM netlist parser, the node name with wildcard character is thus required to be specified as v(node_name) or 'node_name'. Each low-state pulse (the time period when the node stays at logic '0' state) is required to be greater than *low_min_time* and less than *low_max_time*. Each high-state pulse (the time period when the node stays at logic '1' state) is required to be greater than *high_min_time* and less than *high_max_time*.

The values *logic_low_voltage* and *logic_high_voltage* define the threshold voltage of logic '0' state and logic '1' state of the node respectively. The node has a logic '0' state if its node voltage is lower than *logic_low_voltage*; it has a logic '1' state if the node voltage is higher than *logic_high_voltage*.

If the **subckt** parameter is specified, the operation is performed to all instances of the particular subcircuit and the wildcard character '*' can not be used. In addition, the *node_name* needs to be the node name inside the subcircuit, not the full-path names.

Example:

> .**tcheck** check5 **pulsew** data[3] 8n 11n 7n 9n

A pulse width violation error, following the title name of check5, is reported when the low pulse width at node data[3] is less than 8 ns or greater than 11 ns; or when the high pulse width is less than 7 ns or greater than 9 ns.

7.1.4 Timing Edge Check

This command checks the time delay between two specified nodes and reports error when the delay doesn’t meet the specified range. The syntax is

> **.tcheck** *title_name* **edge** *sig_name edge_type ref_name ref_edge_type min_time max_time* <**subckt**=sub_name> <**window**=*window_limit*> <**vlth**=*logic_low_voltage*> <**vhth**=*logic_high_voltage*> <**refvlth**=*ref_logic_low_voltage*> <**refvhth**=*ref_logic_high_voltage*> <**trigger**=0|1|2>

The *title_name* defines the title name of this timing edge check. Any violation resulted from this timing check will be listed in the timing/power check error file, and starts with this title name. The name specified by *sig_name* defines signal node name(s), which can be the node name of a single node or a node name containing wildcard character ‘*’ representing a group of node names. Since the wildcard character ‘*’ is also treated as a comment character in SPICE syntax such that all characters after ‘*’ are treated as comment and ignored by HSIM netlist parser, the node name with wildcard character is thus required to be specified as v(node_name) or ‘node_name’. The name *ref_name* defines the reference node name. *Edge_type* and *ref_edge_type*, that could be one of ‘R’. ‘F’, and ‘X’, define the permissible state transition type for signal node and reference node, respectively: ‘R’ indicates it performs timing check only when the state transition is from ‘0’ to ‘1’; ‘F’ indicates ‘1’ to ‘0’ transition; and ‘X’ indicates any state transition either from ‘0’ to ‘1’ or from ‘1’ to ‘0’. The times specified by *min_time* and *max_time* define the required lower and higher bounds of the timing edge difference between the signal and reference nodes. A timing edge error is reported when the timing edge difference is outside the required range. The timing edge difference is calculated only for the pair of permissible state transitions at the signal node and the reference node. The optional *window_limit* eliminates any timing edge error report if the timing edge difference exceeds *window_limit*.

The values *logic_low_voltage* and *logic_high_voltage* define the threshold voltage of logic ‘0’ state and logic ‘1’ state of signal node respectively. The signal node has a logic ‘0’ state if its node voltage is lower than *logic_low_voltage*; it has a logic ‘1’ state if the signal node voltage is higher than *logic_high_voltage*. The values *ref_logic_low_voltage* and *ref_logic_high_voltage* define the logic threshold voltages for the reference node. When *logic_low_voltage* is specified, *ref_logic_low_voltage* will be defined to the same value if it is not separately defined by **refvlth**. Similarly, when *logic_high_voltage* is specified, *ref_logic_high_voltage* will follow the same value if it is not separately defined by **refvhth**.

If the **subckt** parameter is specified, the operation is performed to all instances of the particular subcircuit and the wildcard character ‘*’ can not be used. In addition, the *sig_name* and *ref_name* need to be the node names inside the subcircuit, not the full-path names.

The optional trigger value defines the condition to trigger the timing edge check: If the value is 0 which is the default, then any permissible state transition at either signal node or reference node will trigger the timing edge check. If the value is 1, then only the permissible state transition at the signal node triggers the check. If the value is 2, then only the permissible state transition at the reference node triggers the check.

Example:

.tcheck check6 **edge** data x ctrl r 2n 5n

When node *data* has a state transition, say at time t2, the time edge difference t2-t1 needs to be within the range of 2 ns and 5 ns, otherwise a timing edge error is reported following the title name of *check6*. The time t1 is the most recent rise state transition time at node *ctrl* before time t2. Moreover, when node *ctrl* has a rise state transition at time t4, the time edge difference t4-t3 needs to be within the range of 2 ns and 5 ns, otherwise a timing edge error is also reported. The time t3 is the most recent state transition time at node *data* before time t4.

Example:

.tcheck check7 **edge** data x ctrl r 2n 5n trigger=2

Same as above example except the edge error check is triggered only by the rise state transition at node *ctrl*. Any edge error is reported following the title name of *check7*.

Example:

.tcheck check8 **edge** data x ctrl r 2n 5n **trigger**=2 **window**=10n

When node *ctrl* has a rise state transition at time t1, an edge error is reported (following the title name of check8) only when t1-t2 is less than 2ns or is less than 10ns but greater than 5ns. The time t2 is the most recent state transition time at node *data* before time t1.

7.1.5 Timing Check Windows

This command defines the timing windows for the timing check commands. The syntax is

.tcheck window *start_time1 <stop_time1 <start_time2 <stop_time2 ….>>>*

All timing check commands use the same set of windows specified by this command. If this command is not specified, the default window for all timing check commands will be from beginning of simulation to the end of simulation. Multiple timing check windows can be specified by providing multiple pairs of start and stop time values, if the stop time is not provided in the last window, it will be extended to the end of simulation.

Example:

.tcheck window 10n 20n 110n 120n 210n

The above command specify three timing check windows, the first window is from 10 ns to 20 ns, the second window is from 110 ns to 120 ns and the third window starts from 210 ns until the end of simulation.

7.1.6 Bisection Optimization

The HSIM simulator supports bisection optimization. The bisection results are printed in the .optz file. Notice that results of the **.measure** and **.print** statements of the last bisection iteration are stored in the .mt and .fsdb/.out files, respectively. Several statements are required to perform bisection optimization:

> **. model** statement,
> **.param** statement, and
> **.tran** statement.

The syntax for the **.model** statement is

> **. model** *opt_model_name* **opt** **method**=<u>*bisection*</u>| <u>*passfail*</u>

The optimization model reference name is specified as *opt_model_name*. The keyword **opt** indicates that this particular **.model** statement is for bisection optimization usage. The same name is used in the corresponding **.tran** statement. Parameter **method** specifies the method to be used in the bisection optimization. A valid value is either <u>*bisection*</u> or <u>*passfail*</u>.

The syntax for the **.param** statement is

> **. param** *param_name=* ***opt****xxx(init_value   low_value   upper_value*)

The name of the parameter used in bisection optimization is specified as *param_name*. **opt***xxx* is the chosen optimzation parameter reference name. The user can replace xxx with suitable choice. The same **opt***xxx* name is referenced in the corresponding **.tran** statement. The initial value, the lower bound, and the upper bound of the parameter are specified as *init_value*, *low_value*, *upper_value*, respectively.

The syntax for the **.tran** statement is

> **.tran** *steptime stoptime* **sweep** **optimize**=**opt***xxx* **result**=*measure_var* **model**=
> *opt_model_name*

The step time and stop time are specified as *steptime* and *stoptime*, respectively. **opt***xxx* is the same optimzation parameter reference name in the corresponding **.param** statement. Parameter **result** specifies the measure variable defined in **.measure** statement. Parameter **model** is to be specified with the same model optimization reference name in the corresponding **.model** statement.

Example:

.**model** *optmod* **opt** **method**=*bisection*
.**param** *delaytime*= **opt***1*(*1n, 0n, 20n*)
.**tran** *0.1n 40n* **sweep** **optimze**= **opt***1* **result**= *maxvout* **model**= *optmod*

7.2 Power Checks

7.2.1 DC Path Check

This command checks the dc current path(s) among voltage sources in the circuit. The syntax is

.pcheck *title_name* **dcpath** <**ith**=*threshold_current*> <*node1* <*node2* ...>> <**period**= *period_time* | ***delay***=*delay_time*> <**start**=*start_time1* <**stop**=*stop_tim1* <**start**=*start_time2* <**stop**=*stop_time2* ...>>>>

or

.pcheck *title_name* **dcpath** <**ith**=*threshold_current*> <*node1* <*node2* ...>> **at**=*t1* <**at**=*t2* …>

Since there may have a large number of dc paths detected by the dc path check command, all reported dc current paths are written into a separate file such that they don't mix with other violation reported by other timing and power checks. Each dc path will be reported in the file *title_name*. The value specified by *threshold_current* defines the threshold current such that each reported dc path current exceeds *threshold_current*, which has a default value of 50 uA. The names specified by *node1, node2, etc*, define the nodes of dc current path. The dc path search starts from any node specified in this node list and ends when it reaches either another node in the list or a dc voltage source node. If no node is specified, then the HSIM simulator reports dc current path(s) between any pair of voltage source nodes. When **period** is specified, the dc path is checked for every *period_time*, starting from the start time of each specified window defined by *start_time* and *stop_time*. If **delay** is specified, then the dc current path is checked at each time t+*delay_time*, where t is the time whenever an input voltage source changes to a new voltage level. In either case when **period** or **delay** is specified, the dc current path is checked only at times within the specified time window(s). If **at** is specified, then the dc current path will be checked at each time directly specified by **at**=*t1*.

Example:

.**pcheck** dc1 **dcpath** **ith**=1e-6 vdd gnd **delay**=5n **start**=10n **stop**=210n

This checks each dc current path between vdd and gnd at every 5 ns after any input voltage source changes its voltage level, and the dc current path is checked only for the time window between 10 ns and 210 ns. The dc current path is reported in the file dc1 if the dc-path current exceeds 1 uA at the time when the dc path is checked.

Example:

> **.pcheck** dc2 **dcpath ith**=1e-6 vcc vss **period**=10n **start**=10n **stop**=210n

This checks each dc current path between vcc and vss at every 10 ns starting from simulation time at 10 ns and simulation time at 210 ns (included). The dc current path is reported in the file dc2 if the dc path current exceeds 1 uA at the time when the dc path is checked.

Example:

> **.pcheck** dc3 **dcpath** vcc vss **at**=130n **at**=150n

This checks each dc current path between vcc and vss at 130 ns time and 150 ns time. The dc current path is reported in the file dc3 if the dc path current exceeds the default value of 50 uA at 130 ns or at 150 ns time.

7.2.2 Excessive Current Check

The excessive element-current check ***exi*** checks whether the current of specified element(s) exceeds a threshold value. The syntax is

> **.pcheck** *title_name* **exi** *elem1* <*elem2…*> <**ith**=*threshold_current*> <**tth**=*exi_time*>
> <**start**=*start_time1* <**stop**=*stop_time1* <**start**=*start_time2* <**stop**=*stop_time2* …>>>>

The *title_name* defines the title name of this excessive current check. Any excessive current report resulted from this check will be listed in the timing/power check error file, and starts with this title name. The excessive current check is applied to each specified element *elem1*, *elem2*, etc. The element name can allow wildcard character '*' such that the excessive current check applies to each element with matching pattern. Since the wildcard character '*' is also treated as a comment character in SPICE syntax, it is required to specify as i(*elem*) or 'elem' whenever the name *elem* contains wildcard character '*'. An element is reported to have excessive current if its element current exceeds *threshold_current* for a time duration longer than *exi_time* specified by **tth**, where *threshold_current* is the current value specified by **ith**. The excessive element current is checked at the time within the time window(s) specified by *start_time1, stop_time1*, *start_time2*, *stop_time2*, etc. If no time window is specified, the check is performed from time 0ns to the end of simulation.

Example:

> **.pcheck** largei **exi** m1 m2 i(x1.*) **ith**=1e-3 **tth**=3n **start**=100n

This checks whether elements m1, m2 and all elements within instance x1 have excessive current larger than 1mA for a time duration longer than 3ns. Any detected excessive-current element after 100 ns will be reported in the timing/power check file *out_file*.chk following the title name of largei. Here out_file is the output file prefix with the default name hsim.

7.2.3 Excessive Rise/Fall Time Check

The excessive rise/fall time check **exrf** checks whether the specified node(s) have excessive rise and/or fall times. The syntax is

> **.pcheck** *title_name* **exrf** *node1* <*node2 …*> <**fanout**=*val2*> <**rise**=*rtime*> <**fall**=*ftime*> <**utime**= u*time1*> <**vlth**=*logic_low_v*> <**vhth**=*logic_high_v*> <**start**=*start_time1* <**stop**=*stop_time1* <**start**=*start_time2* <**stop**=*stop_time2 …*>>>>

Any excessive rise/fall time violation will be reported in the timing/power check file xxxx.chk following the title name of *title_name*. Each name specified by *node1* <*node2 …*> defines signal node name, which can be the node name of a single node or a node name containing wildcard character '*' representing a group of node names. Since the wildcard character '*' is also treated as a comment character in SPICE syntax such that all characters after '*' are treated as comment and ignored by HSIM netlist parser, the node name with wildcard character is thus required to be specified as v(node_name) or 'node_name'.

The optional setting **fanout**=1 limits the rise/fall time checking to those driver nodes with fanouts. This setting will avoid unnecessary check on internal nodes within logic gates. The default **fanout**=0 will check either internal node or driver node. The rise time of the signal is defined as time duration t2-t1, where t1 is the time when the rising signal voltage crosses the voltage *logic_low_v* and t2 is the time when the same continuously rising signal voltage crosses the voltage *logic_high_v*. The fall time of the signal is defined as t4-t3, where t3 is the time when the falling signal voltage crosses the voltage *logic_high_v*, t4 is the time when the same continuously falling signal voltage crosses the voltage *logic_low_v*. The 'U'-state time of the signal is defined as t6-t5, where t5 is the time when the signal voltage enters the 'U' state which is when the signal voltage is between *logic_low_v* and *logic_high_v*, t6 is the time when the signal voltage leaves the 'U' state and the signal voltage is the same as the earlier voltage at t5; in other words, this 'U' state is defined as an incomplete rise or fall transition. The **exrf** check is performed at times within the time window defined by *start_time1*, *stop_time1*, *start_time2*, *stop_time2*, etc. An excessive rise/fall time violation is reported if (1) the signal rise time exceeds *rtime* if it is specified following **rise**, or *utime* instead; or (2) the signal fall time exceeds *ftime* if it is specified following **fall**, or *utime1* instead; or (3) the 'U'-state time exceeds *utime1*. The default value of *rtime*, *ftime* and *utime1* is 5 ns.

Example:

> **.pcheck** longrf **exrf** a1 a2 v(x1.x2.*) **fanout**=1
> \+ **rise**=3n **fall**=4n **vlth**=0.3 **vhth**= 2.7 **start**=100n **stop**=1000n

This checks whether the signal voltages at nodes a1, a2, and those driver nodes which match the pattern of x1.x2.* have excessive rise and fall times. An **exrf** is checked between 100 ns and 1,000 ns and a violation is reported in the timing/power check file following the title name of longrf if the signal rise time exceeds 3 ns, or the signal fall time exceeds 4 ns, or the 'U'-state time exceeds the defaulted 5 ns. The 'U' state is defined as the voltage between 0.3 V and 2.7 V.

7.2.4 High Impedance Node Check

This command checks for high-impedance state in the circuit. The syntax is

> **.pcheck** *title_name* **zstate** *node1 <node2 …>* <**fanout**=*val2*> <**ztime**=z*time1*>
> <**start**=*start_time1* <**stop**=*stop_time1* <**start**=*start_time2* <**stop**=stop*_time2 …*>>>>

Any high-impedance will be reported in the high-impedance check file xxxx.chk following the title name of *title_name*. This check exaimes the high-impedance state of the specified nodes. Each name specified by *node1 <node2 …>* defines signal node name, which can be the node name of a single node or a node name containing wildcard character '*' representing a group of node names. Since the wildcard character '*' is also treated as a comment character in SPICE syntax such that all characters after '*' are treated as comment and ignored by HSIM netlist parser, the node name with wildcard character is thus required to be specified as v(node_name) or 'node_name'. The optional setting **fanout**=1 limits the high impedance state checking to those driver nodes with fanouts. This setting will avoid unnecessary check on internal nodes within logic gates. The default **fanout**=0 will check either internal node or driver node. The **zstate** check is performed at times within the time window defined by *start_time1, stop_time1, start_time2, stop_time2,* etc. Any specified node stays in the high-impedance state longer than *ztime1* will be reported in the timing/power check file following the title name of *name1*. The default value of **ztime** is 5 ns.

Example:

> .**pcheck** highz **zstate** a1 a2 v(x1.x2.*) **ztime**=50n **start**=100n **stop**=1000n

This checks the high-impedance state of the nodes a1, a2 and those nodes matching the pattern of x1.x2.*. The high-impedance state is checked between 100 ns and 1,000 ns. If any specified node stays in the high-impedance state longer than 50 ns, it will be reported in the timing/power check file following the title name of highz.

7.2.5 Power Check Windows

The syntax for power check window is

> **.pcheck window** *start_time1 <stop_time1 <start_time2 <stop_time2 …>>>*

This defines the time window(s) for each power check that doesn't have its own time window specification. Each power check with its own time window specification will not be affected by this **window** setting.

Example:

> .**pcheck window** 10n 20n 110n 120n 210n

This sets three time windows for those power checks without time window specification: from 10 ns to 20 ns, from 110 ns to 120 ns, and from 210 ns to the end of simulation.