

---

# Ambit BuildGates Synthesis Product Notes

Product Version 4.0.8  
May 2001



© 1999-2000 Cadence Design Systems, Inc. All rights reserved.  
Printed in the United States of America.

Cadence Design Systems, Inc., 555 River Oaks Parkway, San Jose, CA 95134, USA

**Trademarks:** Trademarks and service marks of Cadence Design Systems, Inc. (Cadence) contained in this document are attributed to Cadence with the appropriate symbol. For queries regarding Cadence's trademarks, contact the corporate legal department at the address shown above or call 1-800-862-4522.

All other trademarks are the property of their respective holders.

**Restricted Print Permission:** This publication is protected by copyright and any unauthorized use of this publication may violate copyright, trademark, and other laws. Except as specified in this permission statement, this publication may not be copied, reproduced, modified, published, uploaded, posted, transmitted, or distributed in any way, without prior written permission from Cadence. This statement grants you permission to print one (1) hard copy of this publication subject to the following conditions:

1. The publication may be used solely for personal, informational, and noncommercial purposes;
2. The publication may not be modified in any way;
3. Any copy of the publication or portion thereof must include all original copyright, trademark, and other proprietary notices and this permission statement; and
4. Cadence reserves the right to revoke this authorization at any time, and any such use shall be discontinued immediately upon written notice from Cadence.

**Disclaimer:** Information in this publication is subject to change without notice and does not represent a commitment on the part of Cadence. The information contained herein is the proprietary and confidential information of Cadence or its licensors, and is supplied subject to, and may be used only by Cadence's customer in accordance with, a written agreement between Cadence and its customer. Except as may be explicitly set forth in such agreement, Cadence does not make, and expressly disclaims, any representations or warranties as to the completeness, accuracy or usefulness of the information contained in this document. Cadence does not warrant that use of such information will not infringe any third party rights, nor does Cadence assume any liability for damages or costs of any kind that may result from use of such information.

**Restricted Rights:** Use, duplication, or disclosure by the Government is subject to restrictions as set forth in FAR52.227-14 and DFAR252.227-7013 et seq. or its successor.

---

# Contents

---

|   |    |
|---|----|
| <u>Product Notes for Ambit<sup>®</sup> BuildGates<sup>®</sup> Synthesis</u> .....                       | 5  |
| <u>Installation and Licensing</u> .....   | 7  |
| <u>New in Version 4.0.8</u> .....   | 7  |
| <u>Previously Included Installation and Licensing Features in Versions 4.0 to 4.0.7</u> .....           | 7  |
| <u>Online Documentation Delivery System</u> .....   | 7  |
| <u>New in Version 4.0.8</u> .....   | 7  |
| <u>Previously Included Online Documentation Delivery System Features in Versions 4.0 to 4.0.7</u> ..... | 7  |
| <u>64-Bit Compliance on Solaris</u> .....   | 8  |
| <u>New in Version 4.0.8</u> .....   | 8  |
| <u>Product Availability</u> .....   | 8  |
| <u>Executables</u> .....  | 8  |
| <u>Hardware Configuration</u> .....   | 9  |
| <u>Additional Recommended Runtime Settings</u> .....  | 9  |
| <u>Timing Enhancements</u> .....  | 10 |
| <u>New in Version 4.0.8</u> .....   | 10 |
| <u>New Commands</u> .....   | 17 |
| <u>New Globals</u> .....  | 18 |
| <u>Backward Compatibility</u> .....   | 18 |
| <u>Previously Included Timing Analysis Enhancements in Versions 4.0 to 4.0.7</u> .....                  | 19 |
| <u>Test Synthesis Enhancements</u> .....  | 27 |
| <u>New in Version 4.0.8.</u> .....  | 27 |
| <u>Previously Included Test Synthesis Features in Versions 4.0 to 4.0.7</u> .....                       | 32 |
| <u>Low Power Synthesis Option</u> .....   | 34 |
| <u>New in Version 4.0.8</u> .....   | 34 |
| <u>Previously Included Low Power Synthesis Option Features in Versions 4.0 to 4.0.7</u> ..              | 36 |
| <u>The Datapath Synthesis Option</u> .....  | 36 |
| <u>New in Version 4.0.8</u> .....   | 36 |
| <u>Previously Included Datapath Synthesis Features in Versions 4.0 to 4.0.7</u> .....                   | 37 |
| <u>Ambit BuildGates Synthesis Constraint Translator</u> .....   | 38 |
| <u>New in Version 4.0.8</u> .....   | 38 |

## Product Notes for Ambit BuildGates Synthesis

---

|   |    |
|---|----|
| <u>Previously Included Constraint Translator Features in Versions 4.0 to 4.0.7</u> . . . . .  | 38 |
| <u>Verilog and VHDL Enhancements</u> . . . . .  | 39 |
| <u>New in Version 4.0.8</u> . . . . .   | 39 |
| <u>Previously Included Verilog and VHDL Enhancements in Versions 4.0 to 4.0.7</u> . . . . .   | 39 |
| <u>Optimization Enhancements</u> . . . . .  | 41 |
| <u>New in Version 4.0.8</u> . . . . .   | 41 |
| <u>Previously Included Optimization Enhancements in Versions 4.0 to 4.0.7</u> . . . . .       | 41 |
| <u>Distributed Synthesis Enhancements</u> . . . . .   | 44 |
| <u>New in Version 4.0.8</u> . . . . .   | 44 |
| <u>Previously Included Distributed Processing Synthesis Features in Versions 4.0 to 4.0.7</u> |    |
| 46  |    |
| <u>Support of Min Design Rules</u> . . . . .  | 47 |
| <u>New in Version 4.0.8</u> . . . . .   | 47 |
| <u>slibconv Converter</u> . . . . .   | 47 |
| <u>New in Version 4.0.8</u> . . . . .   | 47 |
| <u>Fix List for Version 4.0.8</u> . . . . .   | 48 |
| <u>Fix List for Version 4.0.7</u> . . . . .   | 57 |
| <u>Fix List for Version 4.0.6</u> . . . . .   | 58 |
| <u>Fix List for Version 4.0.5</u> . . . . .   | 60 |
| <u>Fix List for Version 4.0.4</u> . . . . .   | 62 |

---

# Product Notes for Ambit<sup>®</sup> BuildGates<sup>®</sup> Synthesis

---

## May 2001

This product note highlights what is new in this release of Ambit<sup>®</sup> BuildGates<sup>®</sup> synthesis.

**Note:** Refer to the *Product Notes for Cadence<sup>®</sup> PKS* for features and enhancements that apply to the Cadence Physically Knowledgeable Synthesis product.

This release includes the following features and enhancements.

- [Installation and Licensing](#) on page 7
- [Online Documentation Delivery System](#) on page 7
- [64-Bit Compliance on Solaris](#) on page 8
- [Timing Enhancements](#) on page 10
- [Test Synthesis Enhancements](#) on page 27
- [Low Power Synthesis Option](#) on page 34
- [The Datapath Synthesis Option](#) on page 36
- [Ambit BuildGates Synthesis Constraint Translator](#) on page 38
- [Verilog and VHDL Enhancements](#) on page 39
- [Optimization Enhancements](#) on page 41
- [Distributed Synthesis Enhancements](#) on page 44
- [Support of Min Design Rules](#) on page 47
- [slibconv Converter](#) on page 47
- [Fix List for Version 4.0.8](#) on page 48
- [Fix List for Version 4.0.7](#) on page 57
- [Fix List for Version 4.0.6](#) on page 58

## Product Notes for Ambit BuildGates Synthesis May 2001

---

- [Fix List for Version 4.0.5](#) on page 60
- [Fix List for Version 4.0.4](#) on page 62

## Installation and Licensing

### New in Version 4.0.8

There are no new installation or licensing features in version 4.0.8.

### Previously Included Installation and Licensing Features in Versions 4.0 to 4.0.7

The installation mechanism for this release has been changed to the standard Cadence installation methodology called Softload. Please read the `README.txt` file for instructions.

The BuildGates synthesis 4.0 licenses are not backward compatible with previous BuildGates synthesis releases. The new Cadence license manager will not read the older BuildGates synthesis licenses. If you need to continue to run the older BuildGates synthesis releases, you must continue to use the separate BuildGates synthesis license file and license manager.

Please read the `README.txt` file for instructions on configuring the licensing. For information on the standard Cadence licensing procedures, refer to the online document *Cadence License Manager Guide*.

## Online Documentation Delivery System

### New in Version 4.0.8

There are no new on-line documentation delivery system features in version 4.0.8

### Previously Included Online Documentation Delivery System Features in Versions 4.0 to 4.0.7

Ambit BuildGates synthesis 4.0.x uses the updated Cadence online documentation delivery system. The Cadence Online Documentation System Product Notes is provided with the installation software and describes how to use the online system.

## 64-Bit Compliance on Solaris

### New in Version 4.0.8

The 64-bit port overcomes the 4GB (approximately) memory limitation of a 32-bit executable. This allows larger designs to run than were possible—those with approximately 500,000 to one million placeable objects. In fact, the size of the design that can be used is virtually unlimited. Design complexity is the main constraint, but that affects runtime more than anything.

Smaller designs generally do not benefit from using 64-bit, as there is some runtime overhead and increased memory usage simply because the pointer space is 2X.

### Product Availability

The following synthesis, place-and-route products are now 64-bit compliant on Solaris 7 and Solaris 8 systems:

- Ambit® BuildGates® Synthesis
- Cadence® Physically Knowledgeable Synthesis
- Cadence® Datapath Synthesis Option
- Cadence® Low Power Synthesis Option
- Cadence® ultra place (Qplace)
- Cadence® ultra router (Wroute)
- Pearl® timing analyzer
- Cadence® HyperExtract parasitic extractor

### Executables

The 32-bit and 64-bit executables for the tools listed under [Product Availability](#) are installed in the hierarchy, as part of the QSR installation process.

Please refer to the Cadence Installation Guide for more information.

**Note:** The 64-bit executable for Ambit BuildGates synthesis is available in a separate release. Silicon Ensemble™ Place-and-Route, and all other tools used in place-and-route, are only available as 32-bit executables.

## Hardware Configuration

All Sun ULTRA systems and Enterprise servers support 64-bit mode.

To run the 64-bit port, you must be running a 64-bit OS kernel.

## Invoking and Using the 64-bit Executables

When a 64-bit port is available, you can run the 64-bit executables from the installed hierarchy; otherwise, 32-bit versions of the same tools are used. You can select 32-bit or 64-bit executables by setting the following environment variable:

```
CDS_AUTO_64BIT {ALL | NONE | "space-delimited list"}
```

- ALL invokes the 64-bit versions for executables, whenever available.
- NONE invokes only the 32-bit executables.
- "space-delimited list" invokes 64-bit executables for the tools listed and invokes the 32-bit executables for all other tools. This list is a case-sensitive list of executable names, delimited by spaces, semi-colons, or colons. Acceptable values include the following:

```
{ qp | wroute | pearl | hyperExtract | ac_shell }
```

### Example

```
setenv CDS_AUTO_64BIT "wroute"
```

**Note:** Both 32-bit and 64-bit executables run on 64-bit systems. If you do not set this environment variable, the system runs 32-bit executables by default.

## Additional Recommended Runtime Settings

To prevent stack overflow, Cadence recommends setting the maximum stack size available for the process to a higher limit than the typical default values. Setting the stack limit to 16M or higher is recommended, as shown:

```
limit stacksize 32M
```

It is also recommended that you remove any constraints placed on the size of the largest single file allowed in UNIX. Use the following setting to achieve this:

```
limit filesize unlimited
```

## Timing Enhancements

### New in Version 4.0.8

#### False Path Analysis

It is no longer necessary to manually eliminate path violations on false paths in order to find the true critical paths. Timing analysis now identifies the false paths for you. False path analysis considers the functionality of the gates and the circuit connectivity to determine whether a path is a true path or a false path. You can choose either a static (optimistic) or a robust (pessimistic) sensitization criterion for the analysis.

This feature is implemented as additional options to `report_timing` that let you choose the analysis type, the report style, and output file format. False path analysis is done in addition to the analysis specified by any other `report_timing` options.

Critical false path verification is discussed in the Identifying and Eliminating False Paths chapter of the *Timing Analysis for Ambit BuildGates Synthesis and Cadence PKS*. For information about the concepts, see the [Critical False Path Verification Overview](#) section. The sensitization criteria are discussed with examples in the [Static and Robust False Paths](#) section. For information about using the `report_timing -false_path_analysis` options, see the [Reporting and Eliminating False Paths](#) section. Timing report styles are illustrated in the [Example: Timing Reports with False Path Analysis](#) section.

#### Statetable Support

Many vendor libraries contain statetables because statetables provide an easy and understandable method of modeling complex sequential logic such as shift-registers and counters. In releases before 4.0.8, statetables were not supported. A `ff{}` or `latch{}` primitive existing in the same cell definition as `state_table{}` was used instead, if it was defined. Now Ambit BuildGates synthesis and Cadence PKS can understand and map to cells with statetables defined in TLF or `.lib` libraries.

For more information, see [Statetables in TLF Libraries](#) and [Statetables in .lib Libraries](#) in the Using Timing Libraries chapter of the *Timing Analysis for Ambit BuildGates Synthesis and Cadence PKS*.

### **3D Table Support**

Support for three dimensional (3D) lookup tables in Synopsys Liberty (.lib) libraries is available beginning with 4.0.8. Now Ambit BuildGates synthesis supports 3D tables for delay arcs, check arcs, and power arcs in timing libraries.

For examples comparing TLF and .lib 3D tables, see the 3D Table and Design Rule Checks section and the Power Modeling with 1D, 2D and 3D Tables section in the [Examples](#) chapter of the *Timing Library Format Reference*.

### **Minimum DRC for Port Capacitance and Slew Time**

Beginning in 4.0.8, you can specify minimum limits on the design rule checks (DRC) for port capacitance and slew time. By default, the DRCs are taken from the TLF/.lib library if they are present. You can override the library values with the `set_port_capacitance_limit` and `set_slew_time_limit` constraints. Previously, these constraints applied a maximum limit only. The constraints now have a [-min | -max] switch, the default is -max. Also the related globals changed in order to support a minimum limit.

For the complete syntax, see these commands in the *Command Reference for Ambit BuildGates Synthesis and Cadence PKS* :

```
set port capacitance limit
```

The global command `set_global capacitance_limit` is replaced by:

```
set global max capacitance limit
```

```
set global min capacitance limit
```

```
set slew time limit
```

The global command `set_global slew_time_limit` is replaced by:

```
set global max slew time limit
```

```
set global min slew time limit
```

### **Latch Transparency**

A latch is said to be transparent when the input value is allowed to pass through to the output. A latch is opaque when nothing can propagate through the latch.

In 4.0.8, you can make latches permanently transparent or opaque. If a latch is made permanently transparent, the signal on the data pin always propagates to the outputs. Also,

## Product Notes for Ambit BuildGates Synthesis May 2001

---

if there is a constant on the data pin, it is propagated to the outputs. If a latch is made permanently opaque, no signal or constant on the data pin propagates to the outputs.

- You can now make a latch transparent by either:
  - Setting a constant (`set_constant_for_timing`) on the enable pin (or propagating a constant to it).  
All output pins of the latch are transparent.
  - Disabling the arcs (`set_disable_timing`) between the latch enable pin and latch output pins.  
Only the specified latch output pins are transparent.
- To make a latch permanently opaque, you can set a disabling constant on the enable pin or have this constant propagate to the enable pin. All the outputs then become opaque.

For the concepts and terminology, see [Specifying Latch Transparency](#) in the *Timing Analysis for Ambit BuildGates Synthesis and Cadence PKS*. For command examples, see the [Latch Transparency Examples](#) section.

### Limit Time Borrowing

In a latch-based design, time borrowing (cycle stealing) takes advantage of the latch transparency to “steal” time from the next stage to meet timing. By default, time borrowing is enabled and the amount of time borrowed is limited only by the latch clearing clock edge. This 4.0.8 feature lets you set a limit on time borrowing. You can also disable time borrowing for all or part of a latch based design. The time borrow limit assertion can be set on latch cells, enable (clock) pins, and clock waveforms.

For information about the concepts, terminology, and equations used in time borrowing, see [Analyzing Latch-based Designs](#) in the *Timing Analysis for Ambit BuildGates Synthesis and Cadence PKS*. For more information about using the commands, see the [Limiting Time Borrowing](#) section.

For the complete syntax, see these commands in the *Command Reference for Ambit BuildGates Synthesis and Cadence PKS* :

```
set time borrow limit
```

```
get time borrow limit
```

```
reset time borrow limit
```

## **Gated Clocks and Clock Clipping Check**

With the 4.0.8 enhancements for gated clocks you can:

- Specify the treatment of the signal on the clock network. That is, the signal is considered either clock or data depending on how it gets used by the downstream logic. If the downstream logic expects a data signal, set this global:

```
set_global clock_gating_regardless_of_downstream_logic false
```

By default, all pins using the gated clock output are considered clock pins and expect a clock signal (the global is `true`).

- Override the default values for clock gating setup and hold checks with the `set_clock_gating_check` command.
- Check for clock clipping. Clock clipping is a distortion of the clock signal at the gated clock output because the data input was changing or controlling the gate during the time when the clock signal was not controlling the gate. The clock clipping check is a new check for the existing `check_timing` command.

Gated clock concepts are explained in [Using Gated Clocks](#) in the *Timing Analysis for Ambit BuildGates Synthesis and Cadence PKS*. The steps for using a gated clock are given in the [Procedure to Use a Gated Clock](#) section. The checks are described in the [Clock Gating Setup And Hold Checks](#) section and the [Checking for Clock Clipping](#) section

For complete syntax, see these commands in the *Command Reference for Ambit BuildGates Synthesis and Cadence PKS* :

```
set_global clock_gating_regardless_of_downstream_logic
```

```
set_clock_gating_check
```

```
reset_clock_gating_check
```

```
check_timing
```

## **Disable Bus Contention and Floating Checks**

If you know that your design does not have bus contention or floating buses, this enhancement lets you disable the checks by setting these globals:

```
set_global timing_disable_bus_contention_check true
```

```
set_global timing_disable_floating_bus_check true
```

## Product Notes for Ambit BuildGates Synthesis May 2001

---

By default, timing analysis checks for bus contention and floating buses. The default for the globals is `false`.

For more information, see these globals in the *Command Reference for Ambit BuildGates Synthesis and Cadence PKS* :

`set_global timing_disable bus_contention_check`

`set_global timing_disable floating_bus_check`

### Disable Recovery and Removal Checks

Recovery and removal times are often defined for the asynchronous preset/clear pins of registers. This feature lets you disable the checks by setting the following global:

`set_global timing_disable_recovery_removal_checks true`

By default, timing analysis checks for recovery and removal times. The default for the global is `false`.

For more information, see the global in the *Command Reference for Ambit BuildGates Synthesis and Cadence PKS* :

`set_global timing_disable_recovery_removal_checks`

### Clock Report Enhancements

The following `report_clocks` options are new and provide more information about ideal clocks than was previously available:

- `-delay_adjustment_table`
- `-arrival_points`
- `-adjustment_table`
- `-phase_shift_table`
- `-uncertainty_table`
- `-total_shift_table`
- `-insertion`
- `-source_insertion`

## Product Notes for Ambit BuildGates Synthesis May 2001

---

For more information about the options, see [report\\_clocks](#) in the *Command Reference for Ambit BuildGates Synthesis and Cadence PKS* .

### Timing Report Enhancements

Two options have been added to `report_timing` that let you report information about unconstrained paths (paths with no slack).

- `-uncons`

Only the unconstrained paths are reported.

- `-delay_limit float`

Reports only unconstrained paths that are late (or early) relative to delay limit you set.

Also you can now choose the input or output part of bidirectional pins that are in the from/through/to pin list. The following `report_timing` options are new in this release:

- `[-bidi_input_from | -bidi_output_from]`

- `[-bidi_input_through | -bidi_output_through]`

- `[-bidi_input_to | -bidi_output_to]`

For more information about the options, see [report\\_timing](#) in the *Command Reference for Ambit BuildGates Synthesis and Cadence PKS* .

### Storage of RSPF in ADB

Prior to 4.0.8, backannotated detailed or reduced parasitics were not written into the ADB database. You had to read the parasitics file every time you loaded the database. Now the default behavior is to store parasitics (if they are present) in reduced format in the `.adb`. If the parasitics are in detailed format (DSPF) they are reduced and stored as RSPF.

You can choose not to store the parasitics by using the new `-no_spf` option of `write_adb`.

For more information, see [write\\_adb](#) in the *Command Reference for Ambit BuildGates Synthesis and Cadence PKS* .

### SDF TIMESCALE

SDF delays are now automatically scaled if the units given in SDF TIMESCALE and BuildGates units (decided by the first library in target technology list) do not match. You get an Info message with the auto-scaling information.

## Product Notes for Ambit BuildGates Synthesis May 2001

---

If you want to disable auto-scaling, use the `-scale` option of `read_sdf` to specify the scaling factor you want. The delays are scaled by the scaling factor and `TIMESCALE` is ignored. The `-scale` option is not new, but auto-scaling is new for 4.0.8.

For more information, see `read_sdf` in the *Command Reference for Ambit BuildGates Synthesis and Cadence PKS*.

### Faster SDF Writer and Changed SDF Output

Writing SDF for designs with annotated parasitics has been significantly improved. By default, in 4.0.8, `write_sdf` uses delays already cached in the system and writes only fields of the triplets specified by the `pvt_early_path` and `pvt_late_path` globals.

If `pvt_early_path` and `pvt_late_path` are set to `max` (the default for both globals), then only `max` delay values are written and the other two fields are blank (`: :max`). If one is set to `min` and the other is set to `max` as they are for best-case/worst-case analysis, then both `min` and `max` delay values are written (`min: :max`). If the globals are not set, then only the `max` field will be written.

If you want to write out all fields, use the new `-force_calculation` option for `write_sdf`. This option forces recalculation of the delays and writes the full triplet (`min:typ:max`). This was the default behavior prior to 4.0.8.

For more information, see these commands in the *Command Reference for Ambit BuildGates Synthesis and Cadence PKS*:

`write_sdf`

`pvt_early_path`

`set_global pvt_late_path`

`set_global write_sdf_force_calculation`

### Enhancements for Delay Scaling

Now you can set different scaling factors for net delays, cell delays, timing check delays, and SDF assertions. By default the scale factor applies to all three PVT corners, but you can choose to apply the scale factor to one.

The following `set_scale_delays` enhancements are new or changed since 4.0:

- `-net_delay`

## Product Notes for Ambit BuildGates Synthesis May 2001

---

Formerly `-net` in 4.0.

- `-cell_delay`

Formerly `-cell` in 4.0, `-cell` is now equivalent to specifying both `-cell_delay` and `-cell_check`.

- `-cell_check`
- `-incl_sdf`

For complete syntax and examples, see [set scale delays](#) in the *Command Reference for Ambit BuildGates Synthesis and Cadence PKS* .

### Tri-State Handling

In a tri-state gate, the enable pin has the following impact on constant propagation:

- If there is no constant on the enable pin (undefined value X), then the output pin will similarly have no constant, regardless of the logic state of the data pin.
- If logic 1 (or whichever value enables the tri-state gate) propagates to the enable pin, then the logic value of the data pin is transferred to the output. For instance, if the data pin is at logic 0, the output will be at 0 as well. If the data pin does not have any constant, the output will not have any constant either.
- If logic 0 (or whichever value disables the tri-state gate) propagates to the enable pin, then the output is disabled (becomes floating). No signal or constant is propagated from the data pin to the output.

When multiple tri-state gates are connected to a net, conflicting constants should not be driven onto the net. If this happens, timing analysis issues a warning and assumes the undefined value X (no constant) at the sink pins of the net. No new commands are associated with this enhancement.

### New Commands

The following timing commands are new for 4.0.8 and documented in the *Command Reference for Ambit BuildGates Synthesis and Cadence PKS* :

- `get_time_borrow_limit`
- `reset_clock_gating_check`
- `reset_disable_cell_timing`

## Product Notes for Ambit BuildGates Synthesis May 2001

---

- reset disable timing
- reset time borrow limit
- set clock gating check
- set time borrow limit

### New Globals

The following timing globals are new for 4.0.8 and documented in the *Command Reference for Ambit BuildGates Synthesis and Cadence PKS* :

- max capacitance limit
- min capacitance limit
- max slew time limit
- min slew time limit
- timing disable bus contention check
- timing disable floating bus check
- timing disable recovery removal checks
- write sdf force calculation

### Backward Compatibility

There are a few things that you should be aware of if you want to be compatible with the 4.0 release:

- The global `capacitance_limit` is replaced by new globals:
  - max capacitance limit
  - min capacitance limit
- The global `slew_time_limit` is replaced by:
  - max slew time limit
  - min slew time limit
- The default behavior of 4.0 `write_sdf` was to recalculate delays and write all fields of the triplet. In 4.0.8, use the `write_sdf -force_calculation` option to recreate the

## Product Notes for Ambit BuildGates Synthesis May 2001

---

4.0 behavior. Or use the new global. See `write sdf` and `set global write sdf force calculation` in the *Command Reference for Ambit BuildGates Synthesis and Cadence PKS*.

### Previously Included Timing Analysis Enhancements in Versions 4.0 to 4.0.7

**Note:** These timing enhancements also pertain to *Cadence physically knowledgeable synthesis (PKS)*.

#### Enhancement to `set_clock_info_change`

The clock to clock `set_clock_info_change` command on a pin results in starting a new clock signal from the pin. Source and network clock insertion delays from the source clock are transferred to the derived clock. Here, the clock coming into the pin is being called the source clock, and the new clock created on the pin is called the derived clock. Details of how source and network insertion delays are transferred to the derived clock are as follows:

- From the source and derived clock polarities, we first determine edge correspondence i.e., which edge of the derived clock is derived from which edge of the source clock. Edge correspondence is determined as follows:
  - Clock polarity (i.e., positive or negative clock) of the source clock refers to the polarity of the clock signal arriving at the pin. Similarly, clock polarity of the derived clock refers to the polarity of the clock signal being generated from the pin.
  - If source and derived clocks are both of positive polarity, or both of negative polarity, then rising (falling) edge of the derived clock is assumed to be derived from the rising (falling) edge of the source clock. Similarly, if source and derived clocks are of opposite polarity, the rising (falling) edge of the derived clock is assumed to be derived from the falling (rising) edge of the source clock.

Source and network insertion delays from the source clock edges are transferred to the corresponding derived clock edges, if the following two additional constraints are satisfied:

- Derived clock period must be an integral multiple ( $n$ ) of the source clock period, with  $n \geq 2$ .
- For the corresponding edges of the source and derived clocks, following relation must be satisfied:

```
Edge_time_of_derived_clock =  
Corresponding_edge_time_of_source_clock + n *  
period_of_source_clock
```

where  $n$  is an integer.

## Pin Signals

When `set_clock_root` or `set_clock_arrival_time` assertions are present on an internal circuit pin, all incoming (data and clock) signals are blocked from forward propagation. Only the signal asserted on that pin propagates forward.

For `set_clock_arrival_time` assertion, if you use the `-late` option, then only the late incoming signals are blocked. All early signals flow through the pin. Similarly, if you use the `-early` option with `set_clock_arrival_time`, all early incoming signals are blocked, and late signals flow through.

## Path Delay Constraint

You can specify a maximum or minimum delay for a path using the `set_path_delay_constraint` command. The `set_path_delay_constraint` command is equivalent to the `set_max_delay` and `set_min_delay` commands from `dc_shell` and `pt_shell`.

- To specify a path delay constraint, use the `set_path_delay_constraint` command as shown in the following example:

```
set_path_delay_constraint -from I_block/A_reg/Q 10
```

where all paths from `I_block/A_reg/Q` have a maximum time limit of 10 to reach their destination.

**Note:** Before using the `set_path_delay_constraint` command, the `path_style_timing_constraint` global must be set to `true` (default value).

## Path Exception Reporting

The `report_path_exceptions` command gives you a report of all path exceptions in a table indicating whether each path exception is honored or rejected. The timing analyzer follows a priority order for honoring or rejecting path exceptions. Path exceptions are specified using the following commands: `set_false_path`, `set_cycle_addition`, and `set_path_delay_constraint`.

- To generate a report of path exceptions, type the following command in `ac_shell`  
`report_path_exceptions`

Each path exception is listed per line, and each line corresponds to an assertion that you have made. Any wildcards are expanded, and all pins are listed. Clocks are enclosed in quotes.

## Product Notes for Ambit BuildGates Synthesis May 2001

---

Each column labeled From | Through | To corresponds to the `-from`, `-through`, `-to` used to assert the exception.

### Support for Parasitics

Cadence timing analysis has been enhanced to support backannotated interconnect parasitics. Parasitics can be specified in two formats, the Standard Parasitics Format (SPF) and the Standard Parasitics Exchange Format (SPEF). Both file formats are supported. Parasitics are of two forms, the *detailed* and the *reduced*. BuildGates synthesis supports both forms.

The currently supported *reduced* form is composed of a PI-circuit which models the driver load and an Elmore delay for each interconnect source-sink pair. Internally, the delay calculator supports only the *reduced* form which means that all detailed form files read are reduced and the resulting *reduced* form is stored. You can write out the stored *reduced* form with the `write_spf` command. See [write\\_spf](#) in the *Command Reference for Ambit BuildGates Synthesis and Cadence PKS* .

The delay calculator uses the reduced form to compute the gate delay and interconnect delay. The gate delay is computed using an effective capacitance algorithm. The interconnect delay is computed using the Elmore delay and the waveform at the driver.

**Note:** Parasitics stitching is not supported. Parasitics for each flat net must be specified in its entirety in the file. Parasitics for different nets may be specified in different files.

Below are the general steps you use for gate-level timing analysis with parasitics, and/or fixing timing violations after the loading of parasitics.

1. Read the libraries.
2. Read the gate level netlist.
3. Set timing constraints and design rules.
4. Read the parasitics files. For command options, see [read\\_spf](#) and [read\\_spef](#) in the *Envisia and Ambit Synthesis Command Reference*.
5. Perform timing analysis and/or optimization.

### Support for TLF4.3 Libraries

The main advantage of using Timing Library Format (TLF) libraries is that you can use the same library throughout the synthesis place and route (SP&R) design flow. Contact your Cadence AE for more information about when and how to start using TLF libraries.

## Product Notes for Ambit BuildGates Synthesis May 2001

---

Support for TLF libraries includes:

- TLF 4.3 libraries now include support for synthesis, timing, power, and test.

### *Important*

Older versions of TLF are not supported. You must use TLF 4.3 (or later version). Failure to use the correct version of TLF can cause unexpected results.

TLF 4.3 libraries can be supplied by your ASIC vendor or you can use the `syn2tlf` converter described below to convert from a `.lib` format library.

- `syn2tlf` converter

The `syn2tlf` utility converts a Synopsys Liberty (`.lib`) format library to a TLF library. `syn2tlf` produces TLF v4.1 libraries by default. You must use the `-version 4.3` option to produce TLF v4.3 libraries.

### *Important*

You must use the version of `syn2tlf` shipping with this product (or later `syn2tlf` version).

- The TLF power construct `TOTAL_ENERGY` is supported in this release. Energy values described using an average table format for `TOTAL_ENERGY` can be read into LPS and the power can be computed based on that information. See the [Timing Library Format Reference](#) for details about this construct or table format.

## Support for Stamp Models

The Stamp modeling language is a Synopsys proprietary language developed specifically to describe timing models of large custom blocks, such as RAMs and microprocessor cores. Cadence timing analysis in the Ambit BuildGates synthesis and Cadence PKS tools both support Stamp models with the `read_stamp` command.

For a list of supported Stamp features, see [Using Synopsys Stamp Models in Constraint Translator for Ambit BuildGates Synthesis and Cadence PKS](#). Also see [read\\_stamp](#).

## Simultaneous Best Case/Worst Case Analysis

Simultaneous BC/WC analysis lets you analyze the design at two different off chip operating corners. For example, it allows you to fix the setup violations at the worst case operating corner, while fixing hold violations at the best case corner. You can specify the best and worst case operating corner data for various library cells using one library and two operating

## Product Notes for Ambit BuildGates Synthesis May 2001

---

conditions, or two different libraries. For details, refer to the following commands: read\_alf, read\_tlf, set\_operating\_condition, and set\_operating\_parameter in the *Command Reference for Ambit<sup>®</sup> BuildGates<sup>®</sup> Synthesis and Cadence<sup>®</sup> PKS*

The following commands have been added or enhanced for simultaneous BC/WC analysis:

- `set_global timing_analysis_type {min_max | bc_wc}`
- `read_alf or read_tlf -min MinLib -max MaxLib`

For more information and examples, see *Off and On Chip Variation Analysis in Constraint Translator for Ambit BuildGates Synthesis and Cadence PKS*

### Improved Clock and Data Modeling

Enhancements for clock insertion delay (latency) and clock uncertainty include:

- `set_clock` to define ideal clocks (no change)
- The use of the `set_clock_root` command.
- The use of the `set_clock_insertion_delay` command.
- The use of the `set_clock_uncertainty` command.

Enhancement for data specification includes:

- The use of the `set_input_delay` command.

**Note:** The `set_input_delay` command replaces the `set_data_arrival_time` command.

Here are the basic steps for defining clock and data arrival times:

1. Define ideal clock waveforms with `set_clock`.
2. Associate ideal clock waveforms with ports using `set_clock_root` command.
3. Define data arrival times at input ports using `set_input_delay` command, and required times at output ports using `set_external_delay` command.
4. Define clock source and network insertion delay (latency) using `set_clock_insertion_delay` command.

## Product Notes for Ambit BuildGates Synthesis May 2001

---

- a. Any clock source insertion delay is applied in both ideal and propagated modes.
- b. Network insertion delay assertion is honored only during ideal mode. In the propagated mode, network insertion delay is calculated from the actual circuit delays.
- c. Source and network insertion delays specified on the clock waveforms are picked up for both `set_input_delay` and `set_external_delay` assertions.
- d. Along the path from clock port to register clock pins, any insertion delay specification on a port/pin overrides any previous insertion delay on the port/pin/waveform.

For more information, see *Clock and Data Modeling in Timing Analysis for Ambit BuildGates Synthesis and Cadence PKS*

### New Commands

The following timing commands are new:

- `get tech info`
- `read spef`
- `read spf`
- `read stamp`
- `read tlf`
- `report path exceptions`
- `reset tech info`
- `set clock insertion delay`
- `set clock root`
- `set input delay`
- `set path delay constraint`
- `set tech info`
- `write library assertions`

Refer to the *Command Reference for Ambit BuildGates Synthesis and Cadence PKS* for more information.

## Product Notes for Ambit BuildGates Synthesis May 2001

---

### Updated Commands

The following timing commands have been updated:

#### ■ report\_clocks

The `report_clocks` command provides the following new options:

```
report_clocks [-delay_adjustment_table] [-source_insertion] [-insertion]
```

❑ `-delay_adjustment_table`

Reports two tables that detail the early and late path delay adjustments between ideal clocks.

❑ `-source_insertion`

Reports the source insertion delays specified on ideal clocks.

❑ `-insertion`

Reports the network insertion delays specified on ideal clocks. The values in the table are in `min:typ:max` format.

The following option has been modified:

❑ `-adjustment_table`

Reports two tables that detail the early and late cycle adjustments between ideal clocks.

Also, the following options now report two tables (both early and late):

❑ `-phase_shift_table`

❑ `-uncertainty_table`

❑ `-total_shift_table`

#### ■ report\_port

The `report_port` command provides a list of new options to report new timing assertions (input delay, source insertion delay, network insertion delay, clock root, and clock uncertainty). The new options include:

```
report_port -type [input] [source_insertion] [insertion] [clock_root]  
[uncertainty]
```

The `-input`, `-source_insertion` and `-clock_root` options report the assertions in the same format as `-arrival`. An example follows.

## Product Notes for Ambit BuildGates Synthesis May 2001

|          |     |                  |            | Early |      | Late |      |
|----------|-----|------------------|------------|-------|------|------|------|
| Pin Name | Dir | Assertion        | Clock Name | Rise  | Fall | Rise | Fall |
| clk      | IN  | clock_root       | CLK(C)(P)  |       |      |      |      |
| clk      | IN  | source_insertion | *(D)(P)    | 2.50  | 2.50 | 2.50 | 2.50 |
| in       | IN  | input            | CLK(D)(P)  | 3.00  | 3.00 | 3.00 | 3.00 |

The insertion and uncertainty are reported as one value in the following format:

```
insertion          : RISEmin FALLmin : RISEtyp FALLtyp : RISEmax FALLmax
uncertainty       : EARLYrise EARLYfall LATERise LATEfall
```

-to uncertainty is denoted by (T) next to the number. If a value is not specified, it is denoted by - (see below).

| Pin Name | Dir | Assertion   | Value                           |
|----------|-----|-------------|---------------------------------|
| clk      | IN  | insertion   | ( 1.50 3.50 : - - : 2.50 4.50 ) |
| clk      | IN  | uncertainty | 1.00(T) 1.10 - 1.20(T)          |

### ■ set\_clock\_uncertainty

The `set_clock_uncertainty` command has been extended to

- Specify separate clock uncertainty in ideal versus propagated modes.

```
set_clock_uncertainty [-early] [-late] [-ideal] [-propagated]
[-clock_from clk_from] [-clock_to clk_to] [-edge_from {lead | trail}]
[-edge_to {lead | trail}] uncertainty_value
```

- Specify clock uncertainty on a clock pin, affecting uncertainty of paths downstream.

```
set_clock_uncertainty [-early] [-late] [-ideal] [-propagated] -pin
list_of_clock_tree_pins [-rise] [-fall] [-to] uncertainty_value
```

## Product Notes for Ambit BuildGates Synthesis May 2001

---

- set\_global\_clock\_gating\_to\_be\_checked

The default value for `set_global_clock_gating_to_be_checked` has changed from `false` to `true`.

- set\_global\_path\_style\_timing\_constraint

The default value for `set_global_path_style_timing_constraint` has changed from `false` to `true`.

- set\_global\_slew\_propagation\_mode

The default value for `set_global_slew_propagation_mode` has changed from `fast` to `worse_slew`.

- write\_gcf\_assertions

The `write_gcf_assertions` syntax has changed to:

```
write_gcf_assertions [-version {1.3 | 1.4}] [file_name]
```

The default value for the `-version` option of `write_gcf_assertions` has changed from 1.3 to 1.4.

Refer to *Timing Analysis for Ambit BuildGates Synthesis and Cadence PKS* for more information on Timing Analysis.

## Test Synthesis Enhancements

### New in Version 4.0.8.

Refer to *Test Synthesis for Ambit<sup>®</sup> BuildGates<sup>®</sup> Synthesis and Cadence<sup>®</sup> PKS* for more information about test synthesis.

**Note:** These test synthesis enhancements also pertain to Cadence PKS.

### Test Synthesis of Clocked-scan, Clocked-LSSD or Auxiliary Clocked-LSSD Scan Styles.

In addition to muxed-D scan flip-flops, the following types of scan flip-flops are handled:

- clocked-scan
- clocked-LSSD
- auxiliary clocked-LSSD

## Product Notes for Ambit BuildGates Synthesis May 2001

---

The command for clocked-scan style is:

```
set_clocked_scan_clock
```

The commands for clocked-LSSD style are:

```
set_lssd_scan_clock_a <port or internal pin>  
set_lssd_scan_clock_b <port or internal pin>  
set_lssd_aux_clock <port or internal pin>
```

To specify the scan style, use `set_scan_style <muxscan | clocked_scan | clocked_lssd | aux_clocked_lssd>`.

Only edge-triggered flip-flops are converted into scan flip-flops. The following scan flip-flops are not handled:

- Single Latch LSSD
- Double Latch LSSD

### Insertion of Data Lockup Latches Between User-Specified Compatible Clock Domains

Please see the following multi-domain/single chains with data lockup latch insertion:

```
set_scan_style muxscan  
set_scan_mode sen1 1  
set_scan_mode sen2 1  
set_scan_data -clock clk1 sdiRc1 sdoRc1 -enable sen1  
set_scan_data -clock clk2 sdiRc2 sdoRc2 -enable sen2  
set_scan_data -clock clk1 -fall sdiFc1 sdoFc1 -enable sen1  
set_scan_data -clock clk2 -fall sdiFc2 sdoFc2 -enable sen2  
set_dft_compatible_clock_domain clk1 -rise clk2 -rise  
set_dft_compatible_clock_domain clk1 -fall clk2 -fall
```

In this example, the test synthesis tool creates two scan chains since the user has specified clock domain merging using the `set_dft_compatible_clock_domain` commands. The test synthesis tool will insert data lockup latches between the scan chain segments.

Chain 1: scan chain segment clocked by rising edge of `clk1` separated by a data lockup latch to a scan chain segment clocked by rising edge of `clk2`. The scan chain will be associated to scan mode signal `sen1` and scan data ports `sdiRc1` and `sdoRc1`.

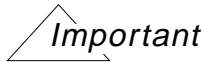
Chain 2: scan chain segment clocked by falling edge of `clk1` separated by a data lockup latch to a scan chain segment clocked by falling edge of `clk2`. The scan chain will be associated to scan mode signal `sen1` and scan data ports `sdiFc1` and `sdoFc1`.

## Product Notes for Ambit BuildGates Synthesis May 2001

---

Scan mode signal `sen1` is the default scan mode signal, specified by the first use of the `set_scan_mode` command.

### PKS Enhancements to Test Synthesis



The following features require a Cadence PKS software license.

#### PKS Ordering of Scan Chains

PKS provides physical placement and timing information along with logic synthesis to resolve the timing closure issues that arise during conventional synthesis. Using placement information from PKS, the test synthesis tool orders and connects the scan chain according to the proximity of its constituent flip-flops. The following commands have options that are related to PKS ordering and connection of scan chains:

##### ■ `do_place -scan_reorder`

This command disconnects existing scan chains, but records the constituent flip-flops of each chain. The command reorders the flip-flops by using placement information from PKS and reconnects the flip-flops in that order. It does not create any new ports.

You can run this command only from the top-timing module. The test synthesis tool connects the scan chains to their respective scan-data in and scan-data out ports as primary inputs at the top level of the design hierarchy.

The `-scan_reorder` option is available only with the `do_place` command. It is not available with the `do_optimize -pks` command.

##### ■ `do_xform_connect_scan -hier -pks -preserve_config`

This command fully configures scan chains, then uses placement information from PKS to order and connect the chains. This command uses existing scan ports, as specified by `set_scan_data`, but it may create ports. Test synthesis will place any newly created ports.

Configuration and chaining of the scan chains occurs relative to the current module (the module specified by a `set_current_module` command).

Your use of the `-preserve_config` option depends on your flow. That is, you run this command after placement. Test synthesis uses the analyzed configuration of the existing scan chains and orders the scan register instances within the chains based on their physical placement with PKS. The analyzed configuration takes precedence over the default configuration or any configuration that you have specified with the `set_number_of_scan_chains` or the `set_max_scan_chain_length` commands.

## Product Notes for Ambit BuildGates Synthesis May 2001

---

You have several options for using test synthesis commands with PKS. The following sections describe how to use test synthesis commands with PKS for RTL designs and mapped scan netlists.

### PKS Reordering of Scan Chain Segments Separated by Data-lockup Latches

The tool can identify existing lockup latches in an existing scan chain. The existence of lockup latch in a scan chain implies that there are multiple clock domains in that scan chain. Hence, PKS reordering will reorder each segment (separated by lockup latches) based on placement information.

#### *Important*

If you are mixing clock domains without any lockup latch, and you do not run `check_dft_rules`, then the tool has no way of determining what belongs to different clock domains. If PKS reordering is run, flip-flops belonging to multiple clock domains are inter-mingled, potentially causing the scan chain to not operate incorrectly.

If you know that there are multiple clock domains in the same scan chain that are not separated by lockup latches, you must run `check_dft_rules` (after specifying the correct test mode setup) so that the tool can correctly identify the clock domains of each flip-flop in the scan chain.

### Multiple Scan Enables in Single Clock Domain/Multiple Chains or Multi-clock Domain Designs

To handle multiple scan enable signals, specify the scan enable signal together with the `scan-in` and `scan-out` signals. To do this, use the following command:

```
set_scan_data <scan_in> <scan_out> [-enable <scan_en>] [-clock <clkname>]  
[-shared_out]
```

When you specify a scan enable signal, this signal is used to connect the `scan_en` pin of the flip-flops in that chain. This occurs between `scan_in` and `scan_out`. If you specify a clock option, this information may be used. When multiple clock domains are merged into a single scan chain, this information is ignored because there is no way to use it. You can specify the polarity of the signal using the `set_scan_mode` option. This option is required if you specify the `scan_en` with the `set_scan_data` command, as shown above.

<scan\_in> and <scan\_en> can be input ports of the current module or an internal signal such as an output of a lower module or instance. In the latter case, a hierarchical reference to the pin of the lower level module and/or the pin of the instance must be specified.

<scan\_out> can be an output port of the current module or an input pin of a lower level

module or instance. In the latter case, a hierarchical reference to the pin of the lower level module and/or the pin of the instance must be specified.

### **Shared Scan-data In with Functional Input Ports and Shared Scan-data Out with Functional Output Ports through Insertion of a Multiplexor**

The specification of shared scan-in and shared scan-out signals is supported by test synthesis. The term, shared scan-in, refers to the usage of a functional signal as the scan data input to the first register in the scan chain. The term, shared scan-out, refers to the multiplexing of the scan data output signal from the last register in the scan chain with the internal functional signal; where the select control signal to the multiplexor is the scan mode signal. The shared scan-in signal is specified directly as the `scanDataInput` argument to the `set_scan_data` command. The shared scan-out signal is specified as the `scanDataOutput` argument and declaring the `-shared_out` option to `set_scan_data` command.

### **Specifying Internal Scan Mode, Scan-data, and Test Mode Signals.**

This release of test synthesis supports the specification of scan mode and scan-data signals to ports of lower level modules or internal pins of technology components in the design database with the following restriction:

The hierarchical path to the internal port or pins for scan-in and scan-out signals is restricted to one level of design hierarchy below the module specified by the `set_current_module` pointer.

#### *Important*

Remember that the `set_current_module` pointer is the module reference in the design hierarchy from which the scan chain connection engine is run.

### **Specifying an Internal Clock Domain as a Separate DFT Clock Domain**

If there are different internal clock domains that are logically connected to the same top level clock in test mode, these internal clock domains can be identified as separate DFT clock domains for scan chain connection purposes. It is a good idea to do this because different clock branches derived from the same logical clock source may have large differences in clock skew. Hence, intermixing flip-flops along different clock branches into the same scan chain can lead to reliability problems during scan shift cycle of the test process. The command `set_dft_internal_clock_domain` can be used to identify an internal clock domain point as a separate DFT clock domain:

```
set_dft_internal_clock_domain \
```

## Product Notes for Ambit BuildGates Synthesis May 2001

---

*<HierObjectIDofLevelModulePortorTechnologyPin>*

### Other New Features

The following new features are described in the Test Synthesis for Ambit BuildGates Synthesis and Cadence PKS document.

- Single-pass test synthesis
- Multiplexed flip-flop scan style
- Top-down and bottom-up scan insertion
- Verilog designs and VHDL designs
- Multiple clock domains
- Multiple balanced scan chains

### Unsupported Features

Test synthesis does not provide solutions for the following:

- Automatic test pattern generation (ATPG)
- Boundary scan
- JTAG
- BIST

### Previously Included Test Synthesis Features in Versions 4.0 to 4.0.7

#### Enhanced Rule Checking

Enhancements have been made to support test design rule checks on generic, mapped, or mixed netlists.

Scan rule checking can analyze under test mode condition.

- Use `set_test_mode_setup` command to specify test mode condition.

## Product Notes for Ambit BuildGates Synthesis May 2001

---

### Automatic Replacement of D-FF to Muxed-D Scan FF during Scan Connection

Enhancements to replace D-FF to muxed-D Scan FF include:

- Flip-flops that pass scan rules are converted automatically to scan FF.
- It is not required to fix any DFT rule violations. If the affected FF are not required to be in a scan chain, the affected FF are automatically excluded from scan. You can use `set_must_scan` to override DFT rule violations, but it is not recommended as the resulting scan chain may not work properly.
- The `set_dont_scan` command is used to avoid DFT error messages or to exclude a FF from scan even if it passes TDRC (optional).
- The `check_dft_rules` command is the scan commit point. This command checks for DFT rule violations, such as gated clocks, derived clocks, and uncontrollable asynchronous signals such as resets.

This information is *sticky* in that the design is stored in memory if the database is saved. The result of the last run determines scan FF and connection. If you make scan rule fixes on some of the flip-flops or want to avoid some of the flip-flops from the scan chain, run the `check_dft_rules` command again.

### Stitching Pre-Scanned Blocks in Higher-Level Scan Chains

Scan chains in lower level modules can be stitched in higher level scan chains as single entities.

### New Commands

The following test synthesis commands are new in this BuildGates synthesis release:

- `do_xform_connect_scan`
- `report_dft_registers`
- `set_dft_transparent`
- `set_dont_touch_scan`
- `set_must_scan`
- `set_test_mode_setup`

Refer to the *Command Reference for Ambit BuildGates Synthesis and Cadence PKS* or more information about the test synthesis commands.

## Obsolete Commands

The following test synthesis commands are obsolete in this release:

- `do_xform_add_test_logic`
- `check_dft_assertions`
- `get_top_dft_module`
- `set_top_dft_module`
- `set_dft_controllable`

## Low Power Synthesis Option

The Cadence<sup>®</sup> low power synthesis option (LPS) requires additional licensing and works in conjunction with BuildGates synthesis.

The LPS option answers the demands to reduce the average power consumption in today's complex designs. The tool identifies how much power is consumed by a design and effectively reduces it without compromising overall chip performance.

Refer to *Low Power Option of Ambit<sup>®</sup> BuildGates<sup>®</sup> Synthesis* and Cadence<sup>®</sup> PKS for more information.

## New in Version 4.0.8

### Enhanced Capabilities to Support DFT

- The flexibility to apply controllability and observability logic on different clock domains or test domains.
- The choice to let the tools automatically create the test mode port.
- The flexibility to create observability registers if you do not like the observability port.
- The ability to choose the clock polarity of the observability register. (By default, the observability register has opposite polarity than gating register banks.) By setting the clock gates you can put the observability registers into the same scan chain as the clock gating register banks.
- In latch mode, the ability to choose the location of the observability gate (before or after the gating latch).

## Product Notes for Ambit BuildGates Synthesis May 2001

---

**Note:** See the `set_clock_gating_options` command in the Cadence® Synthesis Command Reference for the new DFT-related options.

### 3-D Power Table Support

Support for power models with 3D look-up tables in Synopsys .lib. The third dimension of the power template could be either `total_output2_net_capacitance` or `equal` or `opposite_output_net_capacitance` in synopsis .lib format.

### CTPKS Enhancement

A new `-power` option for the `do_build_clock_tree` command. The generated clock tree saves power with the `-ct_no_leaf` option to `do_build_clock_tree`.

**Note:** Currently, the LPS and CTPKS enhancements use the Wire Load Model (WLM) instead of committing the gating logic based on the physical delay and capacitance information from CTPKS, which it will do in a future release. Also, the power estimation and power optimization ignores the glitch power portion.

The LPS option also provides the following features:

### Clock Gating

- Inserts clock gating logic
- Commits during `do_optimize` based on power savings without compromising performance
- Impacts DFT

### Sleep Mode

- Automatically identifies and shuts down datapath elements, functional blocks, and hierarchical modules.
- Commits during `do_optimize` based on power saved and the non-criticality of the signals.

### Gate-Level Power Optimizations

- Power-speedup-based restructuring
- Downsizing of gates

- Buffer removal
- Gate merging
- Slew optimization
- Pin swapping

## **Previously Included Low Power Synthesis Option Features in Versions 4.0 to 4.0.7**

### **Commands**

The following commands and globals are included with the LPS option:

- `report tc stats`
- `do xform optimize power`
- `get power`
- `read_transition_activity_file`
- `report power`
- `set clock gating options`
- `set sleep mode options`

The following LPS options are built into existing BuildGates synthesis commands and are available with a valid LPS license:

- `do_optimize -power`
- `do_xform_optimize_slack -power`

## **The Datapath Synthesis Option**

### **New in Version 4.0.8**

There are no new datapath features in version 4.0.8.

## **Previously Included Datapath Synthesis Features in Versions 4.0 to 4.0.7**

The Cadence® datapath synthesis option requires additional licensing and works in conjunction with BuildGates synthesis and PKS tools.

The datapath synthesis option performs complex arithmetic operations that manipulate data in the RTL (in Verilog or VHDL form) to aid in the development of sophisticated, high-performance ASICs. Arithmetic components such as adders, subtractors, multipliers, comparators, and shifters are used to define the mathematical properties of the design.

Refer to *Datapath Option of Ambit BuildGates Synthesis and Cadence PKS* for more information.

The datapath synthesis option provides the following features:

- Automatic Partitioning of Datapath and Control Components
- Automatic Operator Merging
  - Merging sum-of-products, product-of-products, and product-of-sums
  - Relational operators merged
  - Integrated with logic synthesis
- Automatic Architecture Selection
  - On-the-fly component generation
  - Context-driven
    - Operand width and constant input influence the architecture.
  - Timing-driven
    - Faster for critical paths; smaller on non-critical paths
- Enhanced AmbitWare Components
  - Absolute Value
  - Adder-Subtractors
  - 2-Function Comparator
  - 6-Function Comparator
  - Arithmetic Extension
  - Incrementer-Decrementer

## Product Notes for Ambit BuildGates Synthesis May 2001

---

- Multiplier
- Multiplier-Adder
- Pipelined Multiplier
- Pipeline Register-Delay Line
- Squarer
- Vector Adder
- Arithmetic Shift Right
- Binary Encoder
- Decoder
- Logical Shift Left
- Logical Shift Right
- Leading Zero Counter
- Rotate Left
- Rotate Right

## Ambit BuildGates Synthesis Constraint Translator

### New in Version 4.0.8

There are no new constraint translator features in version 4.0.8

### Previously Included Constraint Translator Features in Versions 4.0 to 4.0.7

The Ambit<sup>®</sup> BuildGates<sup>®</sup> synthesis constraint translator automatically translates Synopsys constraints into the BuildGates synthesis constraint format.

In the Synopsys environment, constraints can be specified or derived from a variety of sources. Users can enter hand-written constraints for Design Compiler (DC) using the `dc_shell` language constructs, or using the `pt_shell` language constructs for PrimeTime (PT). Both `dc_shell` and `pt_shell` provide constraint commands along with general scripting capability, including control structures and variables. DC and PT are both capable of

## Product Notes for Ambit BuildGates Synthesis May 2001

---

generating a `write_script` output, which is essentially an unrolled view of the original constraints.

The BuildGates synthesis constraint translator is capable of translating DC or PT `write_script` output (in `dcsh` and `Tcl` formats) into a BuildGates synthesis format constraints file using the `read_dc_script` command. The translator issues detailed error and warning messages if problems are encountered during translation.

The following enhancements have been added:

- `read_dc_script`

The `read_dc_script` command translates the `dc_shell get_lib_pins` command.

- `read_dc_script`

The `read_dc_script` command translates the `dc_shell set_dont_touch` command to `set_cell_property` for `ac_shell`.

For more information, refer to *Constraint Translator for Ambit<sup>®</sup> BuildGates<sup>®</sup> Synthesis and Cadence<sup>®</sup> PKS*.

## Verilog and VHDL Enhancements

### New in Version 4.0.8

There are no new Verilog or VHDL enhancements in version 4.0.8

### Previously Included Verilog and VHDL Enhancements in Versions 4.0 to 4.0.7

Refer to *HDL Modeling for Ambit<sup>®</sup> BuildGates<sup>®</sup> Synthesis* for information about Verilog and VHDL support.

**Note:** These general enhancements also pertain to the Cadence PKS software.

- Verilog enhancements

- Support for Verilog signed

```
module foo (out1, in1, in2, in3);  
    input [2:0] in1, in2;  
    input signed [2:0] in3;  
    output signed [8:0] out1;
```

## Product Notes for Ambit BuildGates Synthesis May 2001

---

```
    assign out1 = $signed(in1*in2) * in3;
endmodule
```

- Support for `while` and `forever` loops
- Support for `disable`
- Equations writer
- VHDL enhancements
  - Support for VHDL aliases
  - Equations writer
- Full EDIF support
- Miscellaneous enhancements
  - Support for command line elaboration

```
do_build_generic -parameters {{L 0} {R 8}} -design foo
```
  - Support for `map_to_module`  
Provides mapping functions to modules.
- New AmbitWare interface
  - Specialized generators  
Complex arithmetic generator ( $x = a + b * c + d$ ) MUX/Gate generators
  - Integrated with Optimization flow  
Provides constraint-driven implementation selection
  - AmbitWare component library  
The ability to create user-defined component libraries
  - AmbitWare commands

```
delete aware component
report aware library
set aware component property
set aware library
set global aware library search order
```

## Optimization Enhancements

### New in Version 4.0.8

There are no new optimization enhancements in version 4.0.8

### Previously Included Optimization Enhancements in Versions 4.0 to 4.0.7

Refer to the *Ambit BuildGates Synthesis User Guide* for information about optimization.

**Note:** These optimization enhancements also pertain to the Cadence PKS software.

- [Improved Optimization Processing](#) on page 41
- [Multi-Bit Register Mapping](#) on page 41
- [PLA Extraction and Minimization](#) on page 42
- [Set and Reset Priority in Register Mapping](#) on page 43
- [New Commands](#) on page 43
- [Updated Commands](#) on page 43

### Improved Optimization Processing

Optimization processing has been improved to:

- Use a single driver at the `do_optimize` level that decides which transformations to call on the critical path
- Not repeat unsuccessful transformations unless the environment (driver, loads) changes
- A 3X average runtime improvement with negligible QOR impact

### Multi-Bit Register Mapping

The ability to map to multi-bit registers is a new feature.

- Criteria for grouping registers
  - The registers comprise a bus in RTL
  - Clock pins driven by the same signal

## Product Notes for Ambit BuildGates Synthesis May 2001

---

- Set, reset, and enable can be driven by different signals for each bit if the library contains multi-bit registers with individual set, reset, and enable pins for each bit.
- Can map to multi-bit scan registers for DFT
- Can map to multi-bit registers with gated clock for power
- To enable this feature use: `set_global map_to_multibit_registers true`

### PLA Extraction and Minimization

Programmable logic array (PLA) extraction and minimization is a new feature in this BuildGates synthesis release. From both a storage and optimization point of view PLA is a more compact and efficient logic representation than a structural netlist in certain cases (constant case statements, for example)..

PLAs are inferred from constant case statements in `do_build_generic`. A separate hierarchy containing a structural netlist corresponding to the PLA is created to maintain connectivity of the design. At the same time, a PLA representation (structural equivalent) is attached as a hidden property to the created hierarchy. The structural netlist is never used for any optimization. Only the PLA representation is used during structuring. After structuring, the created hierarchy containing the structural equivalent of the PLA is removed.

It is important that you do not manipulate the created PLA hierarchy after running `do_build_generic`. Doing this may result in errors or sub-optimal results. No optimization command will allow a created PLA hierarchy to be passed as the top module to work on.

You can check whether or not a module is a created PLA hierarchy by checking for the existence of the `_pla_module` attribute on the module and skipping those that have this attribute. This is especially important in a bottom-up flow.

PLA extraction and minimization also provides the following features:

- `do_build_generic` extracts PLA from Case statements with purely constant assigns.
- The PLA is first minimized separately for compact netlist generation.
- The PLA is then optimized with the rest of the design.
- Up to 10X improvements in area and runtime for some designs with large `case` statements. In the past, some designs with large `case` statements never finished, but now they finish quickly with good results.
- No new commands or directives are needed because extraction and minimization are performed automatically in the synthesis flow.

## Product Notes for Ambit BuildGates Synthesis May 2001

---

### Set and Reset Priority in Register Mapping

Set and reset priority is now handled properly.

- `do_build_generic` generates priority logic outside of registers (based on the RTL descriptions) to resolve conflict when both set and reset are simultaneously active.
- The mapper attempts to integrate the priority logic into the registers based on the available register types in the library.
- Register types are described in the library with statements indicating the values of Q and Qb when both set and reset are active.
  - `clear_preset_var1:L (or H)`  
Indicates Q is 0 (or 1) when both set and reset are active.
  - `clear_preser_var2:L (or H)`  
Indicates Qb is 0 (or 1) when both set and reset are active.
- Can use LL (HH) registers for LH (HL) requirement by using only Q or Qb, but not both.
- Works with the global `map_inversion_through_registers` command.

### New Commands

The following optimization commands are new in this Ambit BuildGates synthesis release:

- `delete_unconnected_ports`
- `do_xform_ipo`

Refer to the *Command Reference for Ambit BuildGates Synthesis and Cadence PKS* for more information.

### Updated Commands

The following optimization commands have been updated in this Ambit BuildGates synthesis release:

- `set_global auto_slew_prop_selection`

The default value for `set_global auto_slew_prop_selection` has changed to true.

## Distributed Synthesis Enhancements

### New in Version 4.0.8

#### 64-bit Support for Distributed

The general rule for distributed processing synthesis is that each remote `ac_shell` is run with the same command line options as the master `ac_shell`. That applies to licensed options like `-datapath` and `-power` as well as the `-large` option, and now also the new `-64` option. The latter controls whether a 64- or 32-bit `ac_shell` is invoked.

While it is a general rule that a remote `ac_shell` should follow the attributes of the master `ac_shell`, it is not necessary to force each remote `ac_shell` to be set at 64-bit if the master `ac_shell` is 64-bit. Although often, the size of partial designs processed remotely can easily be handled by a 32-bit `ac_shell`, there is currently no reliable metric to determine that in advance. But it is reasonable to assume that when a remote host does support 64-bit, the remote `ac_shell` on that host should be set at 64-bit, provided the master `ac_shell` is set at 64-bit.

If a partial design is processed remotely and if it is too large for 32-bits, the 32-bit `ac_shell` will eventually run out of memory. The entire 64-bit distributed processing run will ultimately fail if that partial design does not end up on a 64-bit host with a 64-bit `ac_shell`.

The rule that each remote `ac_shell` must be 64-bit like the master `ac_shell` is too restrictive. Therefore, in order to provide more flexibility, the following new commands and options have been introduced for 64-bit distributed processing synthesis:

- `set_dist_bits`
- `set_global dist_bits p2`
- `set_host_config host bits p2`
- `get_host_info host kind`
- `get_job_info job rbits`
- `set_dist_bits non-negative_power_of_2`
- `list_of_module_names_or_ids`
- `reset_dist_bits list_of_module_names_or_ids`

## Product Notes for Ambit BuildGates Synthesis May 2001

---

### New Globals

- `set_global dist_rlimit`
- `set_dist_rlimit`
- `reset_dist_rlimit`

The following is a new command related to limiting Distributed jobs:

- `check_rlimit`

### Renamed Commands

- `set_weight` is now `set_dist_weight`
- `reset_weight` is now `reset_dist_weight`
- `set_distribution_point` is now `set_dist_point`
- `reset_distribution_point` is now `reset_dist_point`

**Note:** Old commands will be obsoleted in a future release.

### Restarting Distributed Jobs

The new `restart` feature in distributed processing allows remote jobs to be stopped and automatically restarted. Restarting is similar but not the same as “checkpointing” in that a stopped job restarts from the very beginning, not where it left off at the time it was stopped.

Restarting a distributed job must be done by the user, outside the running master `ac_shell`. Due to standard permission restrictions, typically only the user who started the `ac_shell` with the distributed option can restart a job.

The following are new globals related to restarting distributed jobs:

- `set_global dist_restart_signal`
- `set_global dist_restart_embargo`
- `set_global dist_restart_delay`
- `set_global dist_max_restarts`

The following are new commands related to restarting distributed jobs:

- `restart_script`

## Product Notes for Ambit BuildGates Synthesis May 2001

---

- `check_restart`

### Limiting Distributed Jobs

Distributed remote job limiting relies on the `limit` command. The `limit` command allows some CPU time or other limit to be specified for any BG/PKS command. The new `limit` command provides four different limits:

```
limit -cycle n | cpu_time | -sigUSR1 | -sigUSR2
```

The signal limits `-sigUSR1` and `-sigUSR2` are particularly useful when running distributed jobs. Just like the existing limits, signal limits will be handled “synchronously” in that the command being limited finishes orderly (for internal use: when `fnp_is_interrupt_pending()` is called).

### Previously Included Distributed Processing Synthesis Features in Versions 4.0 to 4.0.7

- In addition to supporting the `do_optimize` command, distributed synthesis also supports the following commands:
  - `do_xform_optimize_generic`
  - `do_xform_map`
  - `do_xform_optimize_slack`
- Distributed synthesis supports user-controlled partitioning.
- Distributed synthesis supports identifying a distribution point using the `set_distribution_point` command.
- The following new distributed commands control the distribution of large jobs to ensure that larger jobs are routed to larger machines and do not end up on machines with limited capacity. Jobs may also be sent to specific LSF queues.
  - `set_weight`
  - `reset_weight`
  - `set_weight_batch_option`
  - `get_weight_batch_option`
  - `set_host_config -weight`
  - `get_host_info -weight`

□ `get_job_info -weight`

## Support of Min Design Rules

### New in Version 4.0.8

Cadence now supports the `min_capacitance` and `min_transition` design rules in the library or as specified by the user through the various new introduced globals or commands for setting these design rules. If these design rules exist in the library or are specified by the user, they are honored and fixed during DRV fixing.

`.ALF`s compiled using earlier versions of `libcompile` for libraries containing these Min Design Rules need not be recompiled because these Min Design Rules have been stored in those `.ALF` files as properties. These can be read in to version 4.0.8 and processed without loss of information.

**Note:** This new feature may have some negative impact on runtime, final area and slack. If you encounter worsened results, check whether the library contains Min Design Rules that were previously ignored and are now being honored.

## slibconv Converter

### New in Version 4.0.8

You can now convert Synopsys `.SLIB` format into the `.SYM` format used by the Ambit Buildgates synthesis schematic viewer. Use the `slibconv` utility included with this release to perform this conversion.

You can find information about this utility and how to convert either `.EDIF` or `.SLIB` to `.SYM` format in the Viewing and Schematic Design chapter of the *Ambit BuildGates Synthesis User Guide*.

Documentation for the `slibconv` utility is also included in documentation provided by our schematic view vendor and can be found in `<release_dir>/BuildGates/<version>/symutils/doc/index.html`.

## Fix List for Version 4.0.8

| PCR Number | TITLE   |
|------------|---|
| 256551     | Help for many globals is not listed if you do <code>help set_global</code> .                          |
| 257179     | Buildscan check and report commands cannot be redirected.   |
| 257874     | Required time reported is different with <code>get_timing</code> and <code>repo</code>                |
| 257930     | <code>Check_dft_rules</code> inconsistent across hierarchy boundary.                                  |
| 258001     | <code>Check_dft_rules</code> reports bogus error.   |
| 258077     | <code>do_xform_connect_scan</code> causes unrecoverable.  |
| 268272     | <code>report_timing</code> reports timing on an non-existent path.                                    |
| 270944     | Reading scan file, check and flag all errors and inconsistencies.                                     |
| 271901     | Demo docs, content and usage, refer to 2.x UI and not to 3.0.   |
| 284258     | <code>naming_style</code> equal to <code>vhdl</code> causes error with <code>check_dft_rules</code> . |
| 284958     | BuildScan ignores <code>fix_multiport_net</code> global.  |
| 285300     | <code>report_timing</code> crash due to excessive number of elements.                                 |
| 285334     | The new distributed synthesis command failed.   |
| 287192     | Doc: Incorrect VHDL example.  |
| 292678     | <code>write_gcf_assertions</code> crashes with internal error.  |
| 296115     | Contradictions in the Component Instantiations and Bindings.  |
| 301762     | Could not create scan-order file due to errors.   |
| 304365     | Analyze routine fails for chained module w/LockUp latch.  |
| 306681     | Lockup latch is printed in scan order file.   |
| 308549     | <code>write_gcf</code> crashes.   |
| 308799     | Functional tests to exercise the scan conn w/precomp blks.  |
| 310400     | Name mapping issue in FNP socket regarding bus bits.  |
| 327240     | Libcompile does not recognize <code>three_state_disable_rise</code> .                                 |
| 328101     | Ambit v3021: Navigates cannot handle big buses.   |
| 329657     | BG cannot recognize async reset but DC can.   |
| 330091     | Excessive runtime on design formatter.  |

## Product Notes for Ambit BuildGates Synthesis May 2001

---

- 332555 `report_timing` not consistent in ideal and propagated mode.
- 333636 RSPFs not stored in `.adb` file.
- 335291 Timing window file significantly different than Pearl.
- 336390 Connection Analyze cycle not updated with the latest `scan_mode` info.
- 336829 Memory corruption for large host lists.
- 336831 Memory corruption for large host lists.
- 337755 Errors in Timing Analysis User Guide.
- 338562 Bidirectional pin of pad cell for `report_timing -through`.
- 339056 PKS scan reordering fails when scandata ports are not declared.
- 340264 `report_net -min_fanout` does not parse all hierarchies.
- 340863 Fatal bug in `clib/list (IsItemNewNext & -Prev)`.
- 340865 Mmempool overhead too high.
- 341228 `syn2bg` doc has needless writings.
- 342567 multi-clock domains lockup latch insertion.
- 342571 `get_scan_chain_info -count` does not work.
- 342761 `libcompile` fails on when string on a power arc.
- 343019 `write_scan_order_file` analysis using `scan_path_inv` variable.
- 343297 `do_xform_optimize_generic -help` is wrong.
- 343974 Insertion Delay not presented correctly in `report_timing`.
- 344244 `libcompile -help` does not return usage help.
- 344382 Scan connection errors found for multiple scan enable signals.
- 344591 Timing user guide says info is wrong or missing.
- 344857 Clear usage & definition of `aware_dissolve_width`.
- 344934 Wrong info about `set_path_delay`.
- 344993 Unable to change threshold value with `set_tech_info -cell`.
- 345030 Multiple clock domains lockup latch insertion failed.
- 345032 Multiple clock domains latch insertion with clock list.
- 345179 Scan connection processing `set_dont_touch_scan` block.
- 345185 Scan connection engine not processing connections properly.

## Product Notes for Ambit BuildGates Synthesis May 2001

---

- 345222 Scan chain analysys w/lockup latch fails.
- 345234 Memory error found when `do_optimize -pks`.
- 345376 Not utilizing pre-existing scan data in/out ports.
- 345393 WARNING: Libcompile does not evaluate expression.
- 345404 No latch insertion with gated clock.
- 345550 Internal clock domain not identified by `check_dft_rules`.
- 345588 Scan configuration error with fixed scan segments.
- 345683 Specify `set_scan_data` to functional port without `-shared` adds sim buf.
- 345840 Highlighting of scan chains.
- 346020 `do_optimize` will not complete in v4.0, but will with v3.0.25.
- 346315 scan chain order.
- 346473 AMBIT3.0.8 vs AMBIT4.0 - 7 minutes vs 80 minutes.
- 347404 Flow test of `set_dont_touch_segment` with design in tieback mode.
- 347433 Scan conn engine adding extra muxes with `-shard_out` option.
- 347887 Large Design bus error - `fnp_exit_if_fatal`.
- 348331 Constraint Translator manual problem.
- 348342 `set_scan_data` to internal pins ignoring top level signals.
- 348480 Scan order file and connect run in tieback mode causes segmentation violation.
- 348752 Spurious error message in Distributed.
- 348878 Cannot set same clock polarity for driving registers and observability registers.
- 348987 Scan configuration error when annotating scan order file.
- 349024 `find -noclock` returns bidi pin after `set_clock_root` assertion.
- 349035 Doc error about `aware_multiplier_architecture`.
- 349181 Unmap/remap routine and scan connection engine adding muxes.
- 349194 Scan config at lower level not working due scan registers at top.
- 349197 Memory error found during scan configuration.
- 349640 FBI crash.

## Product Notes for Ambit BuildGates Synthesis May 2001

---

- 349744 Functional tests to validate internal testmode handling.
- 349885 BG `write_sdf` performance 20x slower than Pearl.
- 350253 Incomplete `set_tech_info` option causes core dump.
- 350304 PKS crashes during `report_timing`.
- 350307 Not expect to find LU with `reset_dft_compatible_clock_domain`.
- 350570 The default precision of `write_sdf` should be 3.
- 350652 Wrong control pin set for clock during `clock_gating`.
- 351198 Description of `top_timing_module` in manual is wrong.
- 351398 Connection engine dropping an output bit after rechaining.
- 351572 PKS reordering fails on netlist.
- 351582 Connection engine not allowing LU to be added at top level.
- 351765 Removal of scan chain connection (SI), if chain not in SOfile.
- 351833 PKS crashed with reading pvt waveformtailres multipliers.
- 351839 PKS crashed with reading `transient_res_model` statement.
- 351864 PKS reordering of 2 scan chains fails.
- 351955 `aware_dissolve_width` does not work.
- 351957 PKS crashed with `waveform_tail_res` statement.
- 352299 Add changes to SP&R flow document.
- 352422 Cannot report operating parameters with min-max libraries.
- 352486 Slight error in Test Synthesis User Guide.
- 352660 Crash in `navigates--error: dft tables could not be set`.
- 352710 Incorrect clock gating control logic Test pins ignored.
- 352779 Chain length balancing not done with `set_max_chain_length`.
- 352794 `set_power_stripe_spec -direction` help text, docs incorrect.
- 352890 PKS reordering of mixed edges/domains in single chain fails.
- 352960 `do_xform_connect_scan` fails to completely uniquify.
- 353516 Distributed Synthesis encounters internal inconsistency.
- 353881 `write_gcf` drops `set_input_delay` constraints.
- 353901 Purify ABW problem in stamp static buffers.

## Product Notes for Ambit BuildGates Synthesis May 2001

---

|        |   |
|--------|---|
| 353982 | BG did core dump during Test synthesis for xxxxx library                    |
| 353985 | <code>INSERTION_DELAY</code> only recorded in corner type.                  |
| 354156 | <code>do_xform_connect_scan</code> removed top level input port.            |
| 354161 | Ambit v.4.0-s004 crashes.   |
| 354229 | <code>do_xform_optimize_generic</code> produce different results.           |
| 354403 | <code>read_tlf</code> dies on <code>INTERNAL_ENERGY</code> construct.       |
| 354504 | Clock signal is propagated as data in a clock gating circuit.               |
| 354519 | Scan chain reconfiguration after placement fails.                           |
| 354545 | <code>report_net</code> does not show wireload model name.                  |
| 354548 | Functional test to validate unconnected/tied SE network with PKS.           |
| 354567 | <code>do_optimize</code> : Tool does not map to set/reset flops in library. |
| 354672 | Scan chain configuration after placement fails.                             |
| 354683 | Distributed synthesis fails during timing optimization.                     |
| 354688 | Negative edge triggered leaf pins detected.                                 |
| 354704 | BG Crashes with error: <code>Could not make connection.</code>              |
| 354720 | Single domain/mixed phase configuration (pre and post place).               |
| 354828 | <code>read_library_update -scaling_factors</code> not accepted.             |
| 354900 | Scan configuration not performed when running in PKS mode.                  |
| 354905 | Scan order file parse error due to latch bit notation.                      |
| 355005 | <code>do_optimize -minimize multiple_output</code> causes failure with PLA. |
| 355064 | <code>check_dft_rules</code> SEGV due to mixed vendor core/IO libraries.    |
| 355114 | Ambit v404: <code>do_opt</code> maps one TIELO cells for an output bus.     |
| 355189 | Data lockup latch analysis in netlist with reordering, no tDRCs.            |
| 355190 | <code>set_data_arrival_time</code> without <code>-clock</code> .            |
| 355281 | Scan configuration from top level with fixed SEG w/an LU fails.             |
| 355283 | Scan config of fixed segments with data LUs after placement.                |
| 355478 | Scan chain re-configuration of lower blocks at top level.                   |
| 355606 | Could not annotate power on input port and bus on GUI.                      |

## Product Notes for Ambit BuildGates Synthesis May 2001

---

|        |  |
|--------|--|
| 355941 | Reconfiguration, removal of inverters to non-existent LUs.                               |
| 356015 | BG40 reads TLF 4.3 or TLF4.1?  |
| 356147 | PKS I/O optimizations runtime too high.  |
| 356149 | <code>report_design</code> ignores <code>SLEW_LIMIT</code> at input pin.                 |
| 356154 | Back-annotated SDF value different then original input.                                  |
| 356354 | LU analysis and PKS reordering in structural database.                                   |
| 356513 | Vague message about test port could not being created.                                   |
| 356557 | Functional tests to validate multi-scan enable/multi-domain designs.                     |
| 356563 | Scan reorder fails in DFT-LPS integration flow.  |
| 356570 | Multiple scan mode usage and PKS reordering of a netlist.                                |
| 356665 | Inconsistency in the options of <code>report_timing</code> and <code>set_FP</code> .     |
| 356704 | Enhance <code>write_scan_order_file</code> to use internal sdi/sdo points.               |
| 356711 | Update <code>read_scan_order_file</code> to support int hierarchical pts.                |
| 356731 | Memory errors occur when mapping a previously mapped module.                             |
| 356794 | Mempool problem.   |
| 356809 | <code>set_dont_modify</code> documentation error.  |
| 356816 | Aware globals missing from Command Reference.  |
| 356821 | Scan conn engine in tieback mode uniquifying <code>AWARE</code> modules.                 |
| 357003 | Clock gating cores dumps during RTL exploration.   |
| 357021 | <code>report_net</code> segmentation violation.  |
| 357057 | <code>WIDTH</code> checks error when <code>read_sdf</code> .                             |
| 357198 | <code>uniquifyCopyAssertionCB</code> : Internal error detected.                          |
| 357300 | Cannot trace asynchronous arc if timing checks exist.                                    |
| 357586 | Test mode port created by <code>autotestport</code> overlap with <code>designpt</code> . |
| 357795 | <code>set_port_capacitance</code> sometimes does not work with a port name.              |
| 357933 | Tests to validate internal scan-data pins and PKS reordering.                            |
| 358289 | Help <code>do_xform_remove</code> has <code>-force</code> option.                        |
| 358311 | Effects of preplacement option and maintaining scan configuration.                       |
| 358317 | Invalid ideal clock if <code>set_false_path</code> invalidates all timing.               |

## Product Notes for Ambit BuildGates Synthesis May 2001

---

- 358346 Could not set sleep mode options.
- 358517 Help `write_edif`.
- 358676 Add global `slew_time_limit` to the output of `report_globals`.
- 358857 Name-mapping problem in `write_assertions`.
- 359240 DFT error if `Test_Mode_*` port in RTL and not DEF as testport.
- 359480 Many MEMPOOL-105 errors on `read_library_update`.
- 359689 Detect inverted scan mode polarity and invert as necessary.
- 359710 The `.adb` file does not accurately store SPF values.
- 359726 Clock gating ignores warning from DFT about non-existent test pin.
- 359753 Preservation of `sdi/sdo/scanenable` paths in `preserve_config`.
- 359759 Mapper inverts the clock-gating logic in the power flow.
- 359785 DRV reported on clock net with `-ignore_clknet`.
- 360035 No control pin defined for clock.
- 360111 Single logic cone in one partition causes structuring hang.
- 360234 PKS User Guide doc references the `set_global` placer variable.
- 360255 Functional tests to validate test assertions saved in `.adb` file.
- 360269 `do_xform_connect_scan` incorrectly connects.
- 360302 Segmentation violation in `do_optimize`.
- 360599 `create_blockage` description in docs is wrong.
- 360600 Detect scan mode polarity and wired nets of opposite polarity.
- 360821 `libcompile` fails.
- 361500 `report_clocks -uncertainty_table` mess  
`report_path_exceptions`
- 361540 Scan order file not capturing registers for `-shared_out` database.
- 361541 PKS reordering (`preserve_config`), disconnects shared output.
- 361545 Scan chain reconfiguration of placed database with `shared_out`.
- 361955 PKS segmentation violation with `read_tlf`.
- 362541 PKS `read_rspf` - inconsistent reduced parasitics warning.
- 362633 Tracing of internal SE network from top level not supported.

## Product Notes for Ambit BuildGates Synthesis May 2001

---

- 362953 Crash when `report_timing` after `import_def -update`.
- 362962 Document `set_dont_move`, `reset_dont_move`, `get_steiner_capacitance`.
- 362966 Scan chain config after placement from tb mode with SE network.
- 363201 `libcompile` crashes with segmentation fault.
- 363205 Reset to Q arcs not disabled with `lib_build_asynch_arc`.
- 363242 Prever-config mode stripping out buffers in sdi/sdo path.
- 363299 Distributed Synthesis encounters error.
- 363393 Name mapping issue between FNP and LV.
- 363399 `libcompile` generates memory error MEMPOOL-105.
- 363598 PKS reordering using `preserve_config` disconnects shared out.
- 363643 SO file analysis of lower level module sdo ports.
- 364242 `read_tlf` crashes - ERROR unrecoverable exception BUS.
- 364342 Ensure multiple scan enables are used across same clock domain.
- 364599 Chain configuration including fixed violation with LU exceeding maximum length.
- 364914 Functional test tieback mode -> placement -> config -> mux insert.
- 365191 Support propagation of test mode signals thru black boxes.
- 365274 Scan config segmentation violation on hierarchical database.
- 365350 Make use of `ctx->theleFlag` in TA timing analysis commands.
- 365351 Timing windows file has null design field.
- 365378 Tests to verify test synthesis of core modules in hierarchical database.
- 366481 Segmentation fault during `do_xform_connect_scan`.
- 366504 `read_spf` crashes on xxxx block.
- 366733 Connection engine segmentation violation at level/0 chains.
- 366866 Segmentation violation during `do_xform_connect_scan`.
- 366870 CTPKS cannot recognize clock pin of scan cell as leaf.
- 367158 Doc for `remove_assertions` not updated.
- 367167 Scan enable net not traced to mux select for shared data.

## Product Notes for Ambit BuildGates Synthesis May 2001

---

- 367604      Ensure propagation of `set_dont_scan` through design hierarchy.
- 368024      Coredump - Scan configuration error LU latch instance property already set.
- 368066      Missing options for docs on `set_supply_rails_on_rows`.
- 368167      User guide references unknown command -  
`set_data_budget_time`.
- 368244      Segmentation violation during scan chain connection of placed dsgn with shared-out.
- 368445      Analysis reporting `NULL` endpoint in SDI path.
- 368446      Analysis reporting `NULL` endpoints in SDO path.
- 368447      Data LU not supported wrt internal clock domains.
- 368459      Connection engine not capturing registers to scan order file.
- 369139      `do_optimize` gets stuck.
- 369294      Unable to set up DP generator.
- 369295      Unable to set up DP generator with `.alf` file.
- 369379      BG hangs in DRV fixing.
- 369452      Name mapping problem in BG/elec and paramgr.
- 369695      Segmentation violation during `do_build_generic` (PLA extracting).
- 369767      Scan chain analysis of multi-bit sdo port fails at top level.
- 369847      Top level port bypassing hierarchical sdi pin after SO file.
- 369960      Segmentation violation during `check_dft_rules`.
- 370482      Verifying options to `get_scan_chain_info` command.
- 370496      Internal scan data pins, `shared_out` support and PKS ordering.
- 371100      Segmentation fault occurs during `do_build_generic` for VHDL code.

## Fix List for Version 4.0.7

| PCR Number | Title   |
|------------|---|
| 256754     | No help available for <code>set_max_transition_time</code> .          |
| 311612     | Memory lead during <code>report_timing</code> after reading SPEF.     |
| 343633     | Error: Invalid constant value, doing <code>read_db</code> .           |
| 352182     | <code>ac_shell -cdsdocd @host</code> always prompts unwanted message. |
| 352965     | Could not recognize the output function of the tri-states.            |
| 355738     | Doc set, get, reset <code>tech_info</code> bug.                       |
| 357559     | BG-Pearl RSPF reduction differ.                                       |
| 358109     | Making some name style functions public.                              |
| 358303     | DP option seq faults with a particular TLF library                    |
| 358316     | <code>do_xform_ipo</code> error--An unrecoverable exception occurred. |
| 360599     | <code>create_blockage</code> description in docs is wrong.            |
| 361431     | <code>read_spf</code> should work from pin connectivity.              |
| 361658     | Datapath is not capable to switch operator.                           |
| 361678     | Crash using datapath during operator merging.                         |
| 361681     | Incorrect tri-state cell marking leads to optimization crash.         |
| 361709     | <code>cdsdoc</code> needs to be ON by default in the BG/PKS GUI.      |

**Product Notes for Ambit BuildGates Synthesis**  
**May 2001**

---

## Fix List for Version 4.0.6

| PCR Number | Title   |
|------------|---|
| 257480     | Usage statement for <code>set_cellproperty</code> differs between docs and                            |
| 297903     | SP&R: <code>read_dc_script</code> support of <code>get_lib_pins</code> .                              |
| 309552     | <code>read_vhdl</code> help and usage are missing <code>-aware_library</code> .                       |
| 309560     | Incomplete help and usage for <code>read_verilog -aware_library</code> .                              |
| 327660     | SPnR: <code>set_dont_touch</code> for library cell reference.   |
| 332166     | SPnR: <code>set_dont_touch</code> for cell reference should be <code>set_cell_property</code> .       |
| 344106     | Document <code>report_hier -tcl</code> .  |
| 344448     | <code>set_input_delay</code> does not generate GCF ARRIVAL assertion.                                 |
| 345430     | <code>set_global use_lef_area</code> .  |
| 345722     | PKS errors out while running on test-case.  |
| 346827     | Add <code>-restructure_aware</code> to the <code>do_optimize</code> documentation.                    |
| 347449     | Message AWARE-313 appears after running <code>do_xform_buffer_tree</code> .                           |
| 349264     | <code>set_clock_insertion_delay</code> fails on internal pin.   |
| 349364     | BG-Pearl DSPF reduction differ.   |
| 349528     | SPnR: <code>read_dc_script</code> ignores <code>-rise/-fall</code> for <code>set_clock_trans</code> . |
| 350222     | No documentation entry for <code>set_global hdl_extract_pla</code> .                                  |
| 350461     | PKS timing window with repeated <code>CONSTANT</code> statement.                                      |
| 350560     | Internal error in BG v4.0, but not in BG v3.0.  |
| 352458     | <code>read_spf -ERROR</code> Failed to reduce detailed parasitics.                                    |
| 352853     | Clock gating logic incorrect.   |
| 353219     | <code>do_extract_critical</code> loses annotated delay and RC on path.                                |
| 353224     | <code>set_dont_move</code> is missing in the documentation.   |
| 353517     | PKS with <code>-datapath</code> crashes in <code>do_build_generic</code> .                            |
| 353640     | <code>write_gcf</code> does not translate <code>set_input_delay</code> .                              |
| 353642     | Request for <code>do_analyze_cross_talk</code> command in BuildGates.                                 |

## Product Notes for Ambit BuildGates Synthesis May 2001

---

- 353881      `write_gcf` drops `set_input_delay` constraints.
- 354182      `do_optimize` does not use `TIEHI/TIELO` cells during mapping.
- 354430      Usage for `timing_disable_recovery_removal_checks`.
- 355821      `remove_assertions` does not work for `input_delay`.
- 356726      4.0 generates the wrong netlist.
- 357296      Busnames are not correct after flattening.

## Fix List for Version 4.0.5

| PCR Number | Title   |
|------------|---|
| 279616     | <code>check_timing</code> gives false warnings of clocks which have been disabled by <code>set_false_path</code> .  |
| 336654     | During <code>do_build_generic</code> , PLAs are automatically extracted for constant case statements. This extraction can potentially cause long runtimes for large non-parallel case <code>x</code> or case <code>z</code> statements.   |
| 338501     | The warning message for <code>read_ctlf</code> is changed to WARNING: Support for <code>read_ctlf</code> is being phased out. CTLF is an obsolete library format. Please obtain a TLF source with version 4.3 or higher and use <code>read_tlf &lt;techlib-355&gt;</code> .   |
| 340248     | <code>do_blast_busses</code> cannot solve name conflicts.   |
| 344256     | An internal error occurs during <code>do_xform_map</code> . The error is: Internal Error in characterize, AT not traceable.   |
| 345238     | New PLA extraction fails in <code>do_build_generic</code> unless the <code>set_global _hdl_extract_pla</code> is set to <code>false</code> .  |
| 345411     | FNP-558 messages are created incorrectly.   |
| 345445     | The pragma architecture is inconsistent with the default architectures.   |
| 346805     | BG fails to generate the correct netlist for the comparison of a bit-sliced value to a shifted value.   |
| 346926     | A failure occurs when Boolean operations have only one driven input. This results in the following error: Allocation of memory by 'array-1: -4-2/0' failed! <FNP-319>.  |
| 348972     | <code>set_disable_timing</code> crashes if the <code>-from</code> points are bus ports.   |
| 349034     | If <code>set_clock</code> is used without the period option, the <code>-waveform</code> option is used to calculate the period. The number is rounded to the nearest integer before performing the arithmetic. This affects <code>report_clocks</code> , <code>write_gcf</code> and <code>write_assertions</code> . |
| 349037     | A warning message is needed if <code>ac_shell</code> is invoked without <code>-datapath</code> and with <code>set_global dp_var</code> set.   |
| 349724     | If no clock is specified for <code>report_cell_instance_timing</code> , some arcs are missing.  |
| 349784     | <code>report_clocks -arrival_points</code> is printing incorrect information if more than one clock source is present.  |

## Product Notes for Ambit BuildGates Synthesis May 2001

---

- 350408      `do_build_generic` causes a consistency check, CDFG\_326.
- 350683      When multiple TLF libraries are read, `read_library_update` gets core dump.
- 351428      The `set_drive_resistance` command does not check for existence of a clock specified with the `-clock` option before asserting it on the system.
- 351464      `report_resources` lists incorrect line numbers.
- 351709      After running `do_xform_fix_hold`, `report_timing -early` generates a timing report that contains a negative delay for a cell instance.
- 351881      `do_build_generic` generates a segmentation fault while reading an RTL netlist.
- 352501      `report_timing -through` the output pin of a cell with unconstrained inputs finds extra paths starting at the output pin of a given cell.

## Fix List for Version 4.0.4

| PCR Number | Title  |
|------------|--|
| 333021     | Clock phase shift is calculated incorrectly.   |
| 339798     | BuildGates crashes when reading a.ctlf lib using <code>read_tlf</code> in the IE environment. An appropriate message should be issued if a syntax error is found in .ctlf or .tlf format.  |
| 340057     | <code>write_verilog</code> and <code>write_vhdl</code> generate illegal indexing of scalar object.   |
| 343016     | When the scan connection engine is invoked, all buffers in the fanout of the scan enable port are removed by the connection engine.  |
| 344378     | Incorrect value of phase shift calculated.   |
| 344526     | Internal error during <code>do_xform_map</code> .  |
| 344572     | Change in <code>report_timing -summary</code> use model.   |
| 344584     | Scan chain reordering with PKS of a lower level block fails.   |
| 345688     | Memory allocation error during <code>read_tlf</code> .   |
| 346365     | <code>do_optimize</code> does not fix design rules as expected.  |
| 346519     | When the <code>hdl_ff_auto_sync_set_reset</code> global is set to true and there is a synchronous incomplete assignment to a variable within a nested if or case statement followed by a top-level if statement in which a constant is assigned to that same variable, BuildGates generated incorrect logic (1'b0) for that data input to the DFF associated with that variable. |
| 346932     | Crash on a <code>PDMNet:get()</code> in <code>readDef</code> .   |
| 347367     | <code>set_max_delay</code> is not translated correctly.  |
| 347397     | PKS DFT reordering caused SEGV when chain ports were dangling.   |
| 349231     | Clock signal is propagated as data in a clock gating circuit.  |