1   Introduction: Placement and Routing

Placement and Routing in L-Edit

This volume of the L-Edit user guide describes the automatic standard cell place and route (SPR) and block place and route (BPR) features of L-Edit.

The chapter Placing and Routing Standard Cell Designs on page 2-15 explains the three SPR modules: core generation, pad routing, and padframe generation. It also explains global signal routing, which is used to route as many as two I/O signals independent of other signals. The chapter Standard Cell Library Designer's Guide on page 2-121 provides design rules for creating standard cell libraries.

The chapter Placing and Routing Block Designs on page 2-144 describes BPR design preparation, design initialization, block placement, and routing. This chapter includes an explanation of the Netlist Navigator. The timing analysis and signal integrity tools are outlined in the chapter Timing Analysis and Signal Integrity in BPR on page 2-239.

There is also a chapter (BPR Tutorial on page 2-261) that outlines how to use the several of the main features in the BPR design flow.

Syntax and usage for the following file formats are detailed in the chapter Place and Route File Formats on page 2-322:

- CAP files—an SPR output file that lists the capacitance, area, and length of each node due to routing.

- EDIF files—a netlist format used as input for SPR and BPR.

- TPR files—a netlist format used as input for SPR and BPR.

- SDF file—an SPR and BPR output file that contains interconnect delays due to routing in standard delay format.

Standard Cell Place and Route (SPR)

Standard cell place and route (SPR) is a place and route package for standard cells that can automatically lay out entire chips. It consists of three modules: a core place and route module to generate a core cell, a padframe generator, and a pad route module to connect the padframe with the core cell. You can run these three modules individually or together.
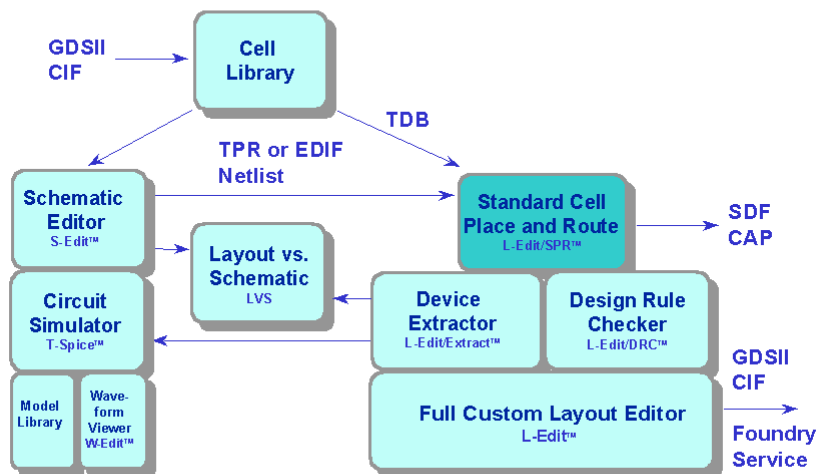
SPR uses standard cells and pad cells from a standard cell library. Netlists can be provided in Electronic Design Interchange Format (EDIF) or Tanner Place and Route (TPR) format. If needed, a mapping tool allows you to achieve consistency between the cell and pin names in your netlist and your library.

SPR generates a core, a padframe, and a chip cell in L-Edit which then can be checked for design rules and extracted. In order to verify the delay constraints, you can generate a nodal capacitance (CAP) file or a standard delay format (SDF) file, or both, during the place and route step.

The place and route steps are fully automated. You can use two- or three-layer routing, with the latter including the option of over-the-cell (OTC) routing. Up to two I/O signals (e.g., clock signals) can be routed separately to better control delay and skew when you use the global input signal routing function. Among the many features of SPR are standard cell grouping (cell clustering) and critical nets consideration.

The following diagram illustrates the design flow for L-Edit/SPR within the Tanner EDA tool suite.

L-Edit / SPR Design Environment

GDSII CIF → Cell Library

TDB

TPR or EDIF Netlist

Schematic Editor
S-Edit™

Standard Cell Place and Route
L-Edit/SPR™

SDF CAP

Layout vs. Schematic
LVS

Circuit Simulator
T-Spice™

Device Extractor
L-Edit/Extract™

Design Rule Checker
L-Edit/DRC™

Model Library

Wave-form Viewer
W-Edit™

Full Custom Layout Editor
L-Edit™

GDSII CIF

Foundry Service

Contents/Search **Index**

Block Place and Route (BPR)

Block place and route (BPR) is an automated block placement and routing tool. You can use it to incorporate intellectual property blocks, custom blocks, standard cells, and SPR-generated cores and padframes into your design, and to perform chip assembly and basic floorplanning.

BPR initializes a design by reading an EDIF or TPR netlist for the blocks to be used and their connectivity, then places instances of those blocks into a top-level cell in your design file.

Once a BPR design is initialized, blocks can be placed and routed either automatically or manually. BPR allows you to perform incremental placement, where the position and orientation of placed blocks will be retained when you add new blocks to your design.

BPR supports fully automatic routing or assisted manual routing. During assisted manual routing, BPR displays routing guides that help you move from pin to pin as you route your nets. Keyboard shortcuts allow you to quickly select from the defined routing layers, and BPR automatically adds the appropriate via when you change from one routing layer to another.

BPR also provides timing analysis and signal integrity analysis of a design at any stage of the BPR process. Running these analyses interactively with placement and routing allows you to evaluate the impact of your chosen placement, driver

and receiver parameters, and interconnect topology on your design. The ability to perform "what-if" analyses allows you to optimize device parameters, block-level topology, and floorplanning without having to complete the physical design, avoiding costly design iterations.

The following diagram illustrates the design flow for L-Edit/BPR in a Tanner EDA tools environment.

L-Edit / BPR Design Environment

```
GDSII          ┌─────────┐              ┌──────────────────┐
CIF  ────────▶ │  Cell   │              │  Standard Cell   │
               │ Library │              │ Place and Route  │
               └─────────┘              │   L-Edit/SPR™    │
                                        └──────────────────┘
                    TPR or EDIF      TDB            │ TDB
                    Netlist                         ▼
  ┌───────────┐                        ┌──────────────────┐
  │ Schematic │ ─────────────────────▶ │      Block       │      SDF
  │   Editor  │                        │ Place and Route  │ ────▶
  │  S-Edit™  │                        │   L-Edit/BPR™    │
  └───────────┘   ┌─────────────┐      └──────────────────┘
  ┌───────────┐   │ Layout vs.  │
  │  Circuit  │   │  Schematic  │◀──┐
  │ Simulator │   │     LVS     │   │
  │  T-Spice™ │   └─────────────┘   │
  └───────────┘  ┌─────────┬────────┴──┐
  ┌─────┬─────┐  │ Device  │  Design   │
  │Model│Wave-│  │Extractor│   Rule    │      GDSII
  │Libr-│form │  │ L-Edit/ │  Checker  │      CIF
  │ ary │Viewer│ │Extract™ │ L-Edit/DRC™│ ────▶
  │     │W-Edit™│ └─────────┴───────────┘
  └─────┴─────┘  ┌──────────────────────┐      Foundry
                 │ Full Custom Layout   │      Service
                 │      Editor          │ ────▶
                 │      L-Edit™         │
                 └──────────────────────┘
```
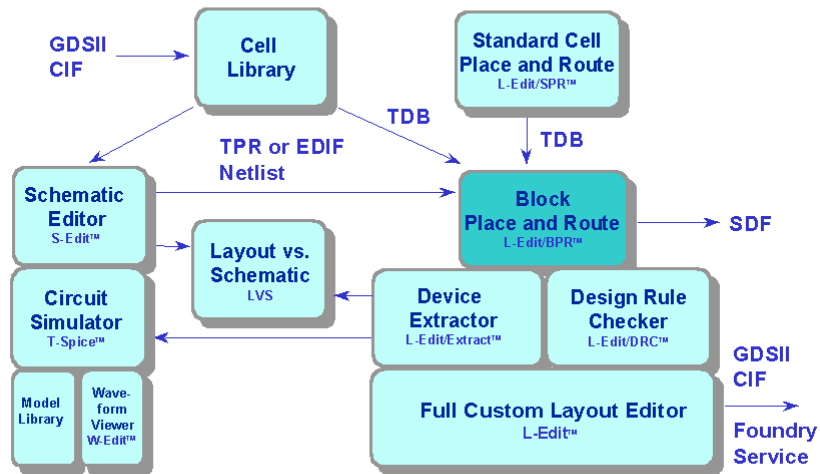
Documentation Conventions

This section contains information about the typographical and stylistic conventions used in this user guide.

Special Fonts

The following inline references are represented by a bold font:

- Menu and simulation commands (For example: **.print tran v(out)**.)
- Literal user input (For example: Enter **14.5**.)
- Program output (For example: S-Edit generates names for the ports on the symbol based on the **PAD** string.)
- All dialog elements—fields, checkboxes, drop-down menus, titles, etc. (For example: Click **Add**.)

Freestanding quotations of input examples, file listings, and output messages are represented by a constant-width font—for example:

```
.ac DEC 5 1MEG 100MEG
```

Variables for which context-specific substitutions should be made are represented by bold italics—for example, ***myfile*.tdb**.

Sequential steps in a tutorial are set off with a checkbox (☑) in the margin.

References to mouse buttons are given in all capitals—for example, MOVE/ EDIT. When a key is to be pressed and held while a mouse button is used, the key and button are adjoined by a plus sign (+). For example, **Shift**+SELECT means that the **Shift** key is pressed and held while the SELECT mouse button is used.

The terms "left-click," "right-click," and "middle-click" all assume default mappings for mouse buttons.

Text omitted for clarity or brevity is indicated by an ellipsis (**…**).

Menu Commands and Dialog Titles

Elements in hierarchical menu paths are separated by a **>** sign. For example, **File > Open** means the **Open** command in the **File** menu.

Tabs in dialog boxes are set off from the command name or dialog box title by a dash. For example, **Setup > Layers—General** and **Setup Layers—General** both refer to the **General** tab of the **Setup Layers** dialog.

Special Keys

Special keys are represented by the following abbreviations:

Key	*Abbreviation*
Shift	**Shift**
Enter	**Enter**
Control	**Ctrl**
Alternate	**Alt**
Backspace	**Back**
Delete	**Del**
Escape	**Esc**
Insert	**Ins**
Tab	**Tab**
Home	**Home**
End	**End**
Page Up	**PgUp**
Page Down	**PgDn**

Key	*Abbreviation*
Function Keys	**F1 F2 F3** …
Arrow Keys	$\downarrow, \leftarrow, \rightarrow, \uparrow$

When certain keys are to be pressed simultaneously, their abbreviations are adjoined by a plus sign (+). For example, **Ctrl**+**R** means that the **Ctrl** and **R** keys are pressed at the same time.

When certain keys are to be pressed in sequence, their abbreviations are separated by a space ( ). For example, **Alt**+**E R** means that the **Alt** and **E** keys are pressed at the same time and then released, immediately after which the **R** key is pressed.

Abbreviations for alternative key-presses are separated by a slash (/). For example, **Shift**+$\uparrow$ / $\downarrow$ means that the **Shift** key can be pressed together with either the up ($\uparrow$) arrow key or the down ($\downarrow$) arrow key.

Online User Guide Conventions

Text	Chapter title
Text or **Text**	Indicates a hypertext link
Contents/Search	Indicates a link to the Table of Contents file
Index	Indicates a link to the Index file
◀ ▶	Indicates a link to the previous/next page

Acrobat Reader Toolbar Tips

Go to previous/next page

Go to start/end of chapter

Go back/forward through pages viewed

Perform a full-text search

Open the search results list

Go to previous/next search result item