Milkyway™ Database Application Note

Version J-2014.09, September 2014

SYNOPSYS®

Copyright and Proprietary Information Notice

Destination Control Statement

All technical data contained in this publication is subject to the export control laws of the United States of America. Disclosure to nationals of other countries contrary to United States law is prohibited. It is the reader's responsibility to determine the applicable regulations and to comply with them.

Disclaimer

SYNOPSYS, INC., AND ITS LICENSORS MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Trademarks

Synopsys and certain Synopsys product names are trademarks of Synopsys, as set forth at https://www.synopsys.com/company/legal/trademarks-brands.html.
All other product or company names may be trademarks of their respective owners.

Free and Open-Source Licensing Notices

If applicable, Free and Open-Source Software (FOSS) licensing notices are available in the product installation.

Third-Party Links

Any links to third-party websites included in this document are for your convenience only. Synopsys does not endorse and is not responsible for such websites and their practices, including privacy practices, availability, and content.

www.synopsys.com

Feedback

Contents

Feedback

About This Application Note

This application note describes the Milkyway™ database, a unifying design storage format for tools in the Synopsys Galaxy™ Design Platform, including Design Compiler®, IC Compiler™, StarRC™, IC Validator, PrimeRail, and the Milkyway Environment.

This application note is for engineers who use one or more Synopsys tools to store or access physical design data in the Milkyway format.

This preface includes the following sections:

New in This Release

Information about new features, enhancements, and changes, known limitations, and resolved Synopsys Technical Action Requests (STARs) is available in the Milkyway Release Notes on the SolvNetPlus site.

Related Products, Publications, and Trademarks

For additional information about the Milkyway database and tools that work with the database, see the documentation on the Synopsys SolvNetPlus support site at the following address:

https://solvnetplus.synopsys.com

Conventions

The following conventions are used in Synopsys documentation.

Convention	Description
`Courier`	Indicates syntax, such as `write_file`.

Feedback

Convention	Description
Courier italic	Indicates a user-defined value in syntax, such as `write_file design_list`
Courier bold	Indicates user input—text you type verbatim—in examples, such as `prompt> write_file top`
Purple	• Within an example, indicates information of special interest. • Within a command-syntax section, indicates a default, such as `include_enclosing = true | false`
[ ]	Denotes optional arguments in syntax, such as `write_file [-format fmt]`
...	Indicates that arguments can be repeated as many times as needed, such as *pin1 pin2 ... pinN.*
|	Indicates a choice among alternatives, such as `low | medium | high`
\	Indicates a continuation of a command line.
/	Indicates levels of directory structure.
Bold	Indicates a graphical user interface (GUI) element that has an action associated with it.
Edit > Copy	Indicates a path to a menu command, such as opening the **Edit** menu and choosing **Copy**.
Ctrl+C	Indicates a keyboard combination, such as holding down the Ctrl key and pressing C.

Customer Support

Customer support is available through SolvNetPlus.

Accessing SolvNetPlus

The SolvNetPlus site includes a knowledge base of technical articles and answers to frequently asked questions about Synopsys tools. The SolvNetPlus site also gives you access to a wide range of Synopsys online services including software downloads, documentation, and technical support.

To access the SolvNetPlus site, go to the following address:

https://solvnetplus.synopsys.com

If prompted, enter your user name and password. If you do not have a Synopsys user name and password, follow the instructions to sign up for an account.

If you need help using the SolvNetPlus site, click REGISTRATION HELP in the top-right menu bar.

Contacting Customer Support

To contact Customer Support, go to https://solvnetplus.synopsys.com.

1

The Milkyway Database

The Milkyway database is the unifying design storage format for the Synopsys Galaxy™ Design Platform. The database provides persistent storage of physical design data that links Galaxy platform tools together. The database is periodically updated with new features to support advances in EDA technology.

The basic features of the Milkyway database are described in the following sections:

- Milkyway Database Overview
- Milkyway Database Versions
- Milkyway Libraries and Cells
- Milkyway-Related Commands
- Physical Library Data Preparation
- Database Naming Conventions

Detailed usage information is provided in the documentation for the respective tools that access the Milkyway database, including the Design Compiler, IC Compiler, IC Validator, PrimeRail, StarRC, and Milkyway Environment tools.

Milkyway Database Overview

The Milkyway database is the unifying design storage format for the Synopsys Galaxy Design Platform. The database provides persistent data storage that links Galaxy platform tools together, which eliminates the need for large, intermediate exchange files and prevents loss of design intent that could occur using other data exchange formats. The Milkyway database format is regularly augmented with new capabilities to support tool features such as signal integrity analysis, power reduction, and yield enhancement.

The Milkyway database offers the following features and benefits:

- Production-proven database for all Galaxy Design Platform tools
- Capacity to support the largest designs

- Support for the latest technology nodes, including 28 nm, 20 nm, 14 nm, and beyond
- Third-party data input to the Galaxy Platform via LEF and DEF

For today's large designs, tremendous volumes of data are generated as designs move through the implementation phase. Without a common database, tools must communicate through ASCII design interchange files that present a range of problems, including excessive files sizes and transfer times, errors due to semantic mismatches, and inconsistent revision of data formats between different tools.

The Milkyway database solves these issues for the Galaxy Platform because all implementation tools have direct, binary interfaces to the database. Direct database access eliminates the need for large, slow ASCII interchange files and semantic gaps between different tool functions. Each tool is able to see the data in the same way, preventing costly errors and reducing design iterations.

The following Synopsys tools use the Milkyway database:

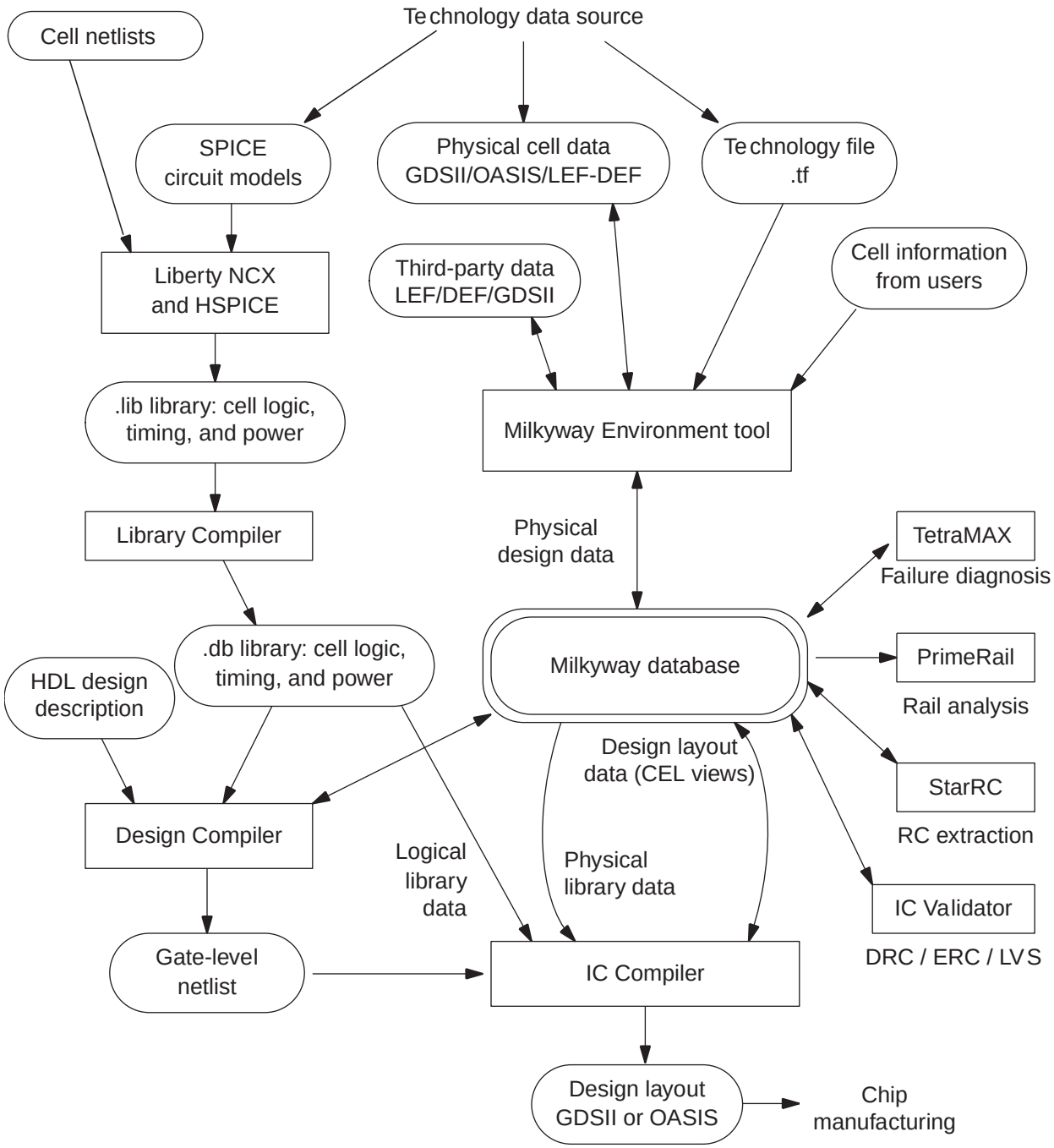
- Design Compiler writes a mapped, uniquified design into the Milkyway database with the `write_milkyway` command, including the netlist and synthesis constraints. It also writes the Synopsys Physical Guidance information, if any. It can also write out the design in other formats, such as .ddc and Verilog.
- IC Compiler reads physical design information and library cell information from the Milkyway database to perform placement, clock tree synthesis, and routing. It writes the resulting chip design information back into the Milkyway database.
- The Milkyway Environment tool prepares new library cells from physical data in other formats, including GDSII, OASIS, and LEF/DEF, and writes the new library cells to the Milkyway database. The Milkyway Environment tool can also be used to copy, edit, and delete cells; and to perform blockage, pin, and via (BPV) extraction to make FRAM views.
- IC Validator reads physical design information from the Milkyway database to perform design rule checking (DRC), electrical rule checking (ERC), layout versus schematic (LVS) checking, and fill-pattern generation. It writes the results back into the database for IC Compiler to read and use for error reporting.
- PrimeRail reads physical design data from the CEL and FRAM views in the Milkyway database to perform IR drop and electromigration analysis. In some cases, it reads connectivity information from the CONN view.
- StarRC reads physical design information from the Milkyway database to perform parasitic RC extraction. It writes the results back into the database for IC Compiler to read and use for timing and crosstalk analysis.
- Older tools such as JupiterXT™, Astro®, and Hercules™ use the Milkyway database in a manner similar to the corresponding current tools.

For detailed information about using Synopsys tools with the Milkyway database, see the documentation for the respective tools, available on SolvNet. The Milkyway Environment tool can be used for Milkyway library maintenance and library cell preparation as described in the *Library Data Preparation for IC Compiler User Guide*, available on SolvNet in the IC Compiler documentation collection.

Figure 1 shows some of the chip design and analysis data flows using the Milkyway database.

Feedback

Figure 1 Milkyway Database Usage in the Galaxy Design Platform



Feedback

Milkyway Database Versions

The Milkyway database infrastructure is periodically updated to support new database features. Each new version of the database is called a “schema.”

Table 1 shows the earliest IC Compiler and Milkyway Environment product version number corresponding to each new Milkyway schema.

Table 1 IC Compiler and Milkyway Environment Versions and Milkyway Schemas

IC Compiler and Milkyway Environment version	Milkyway schema
J-2014.09	8.1
H-2013.03	7.0
F-2011.09	6.0
D-2010.03-SP1	5.1 (compatible with schema 5.0)
D-2010.03	5.0
C-2009.06	4.0
B-2008.09	3.2
A-2007.12	2.0

When IC Compiler or Milkyway Environment version J-2014.09 reads Milkyway data stored under schema 7 or earlier, the tool automatically updates the data to schema 8.1. Milkyway data stored under schema 8.1 is compatible with IC Compiler and Milkyway Environment version I-2013.12-SP3 and later. For example, IC Compiler version I-2013.12-SP3 can read and update Milkyway data stored under schema 8.1. However, versions of tools earlier than that, such as IC Compiler version I-2013.12, cannot read schema 8.1 data. Schema 8.1 data cannot be converted back to an earlier schema.

Table 2 is a list of Synopsys tools and the release numbers of those tools that are compatible with Milkyway schema 8.1.

Table 2 Versions of Synopsys Tools That Support Milkyway Schema 8.1

Synopsys tool name	Tool releases that support Milkyway schema 8.1
Custom Designer	J-2014.12 and later. I-2013.12-SP2 and later can read schema 8.1 data; J-2014.12 uses schema 8.1 by default.

Feedback

Table 2 Versions of Synopsys Tools That Support Milkyway Schema 8.1 (Continued)

Synopsys tool name	Tool releases that support Milkyway schema 8.1
Design Compiler	J-2014.09 and later. I-2013.12-SP4 and later can read schema 8.1 abstraction models.
Formality	I-2013.12-SP4 and later
IC Compiler	J-2014.09 and later. I-2013.12-SP3 through I-2013.12-SP5 can read and write both schema 7 and schema 8.1 data.
IC Validator	I-2013.12-SP2-1 and later
PrimeRail	I-2013.12-SP3 and later for the default layer mode; support for the extended layer mode is planned for a future release
PrimeTime suite	J-2014.06 and later
StarRC	I-2013.12-SP3-1 and later; writes PARA data but not CEL data; no implicit conversion
TetraMAX ATPG	J-2014.09 and later; reads X-Y coordinates of cell nets and topologies only

Note:

The latest versions of Hercules and PrimeYield LCC use schema 5. There are no plans to update Milkyway database support for Astro or JupiterXT. MVRC (MVtools) and Library Compiler do not use the Milkyway database.

You can explicitly convert Milkyway cells and libraries from the previous to the current Milkyway schema by using the `convert_mw_lib` command in the IC Compiler or Milkyway Environment tool.

If you do not convert the data explicitly, when you open a Milkyway cell with the latest version of the tool, the tool performs an implicit conversion of the cell and issues a message about the conversion. You should save the converted cell before you close the library. Otherwise, the conversion will need to be performed again when you reopen the cell.

Using the `convert_mw_lib` command ensures complete conversion of your library data and is more efficient than converting cells one at a time. For more information, see “Milkyway Database Versions” in the *Library Data Preparation for IC Compiler User Guide*.

Milkyway Libraries and Cells

A physical library contains information about the geometry of the cells that are placed in the design and connected with power, ground, clock, and signal routes. This library

information includes the cell dimensions, borders, pin locations, and mask layers, as well as technology information such as wire tracks, antenna rules, and electromigration data.

The physical library information is maintained in the Milkyway database. Synopsys tools can access the design and library information in the database. The database contains not only leaf-level physical cell information and technology information, but also design-specific physical information such as the placement and routing of the design.

The Milkyway database is organized as a hierarchy of data files. However, you must not create, delete, copy, or edit these files directly using operating system commands such as `cp` and `rm`. Instead, you should access the database by using a compatible tool such as IC Compiler or Milkyway Environment and use the tool commands to read, write, or change the database contents. This ensures the consistency and integrity of the database.

In the Design Compiler, IC Compiler, or Milkyway Environment tool, you open a Milkyway database for viewing or editing with the `open_mw_lib` command. By default, opening a Milkyway library makes that library accessible to the tool for both reading and writing physical design information. You can open no more than one Milkyway design library at a time. However, the design can contain references to cells contained in other Milkyway libraries, called reference libraries. Multiple users can open the same design library in different sessions. However, only one user at a time can have permission to write into a Milkyway design library.

The basic unit of information in a Milkyway library is the cell. A cell is a representation of a physical structure in the chip layout, which can be something as simple as an I/O pad or as large and complex as an entire chip. In the IC Compiler or Milkyway Environment tool, you open a cell for editing by using the `open_mw_cel` command. The cell must be contained in the Milkyway library that is currently open.

A chip design is typically built as a hierarchy of cells. The entire chip is a single cell built out of lower-level blocks, which are also cells. These blocks are built out of smaller blocks, and so on, down to the level of leaf-level cells, which are gate-level standard cells.

Cell Views

The Milkyway database can contain different representations of the same cell, called “views” of that cell. These are the main types of views used in physical implementation tools:

- CEL view: The full layout view of a physical structure such as a via, standard cell, macro, or whole chip; contains placement, routing, pin, and netlist information for the cell
- FRAM view: An abstract representation of a cell used for placement and routing; contains only the metal blockages, allowed via areas, and pins of the cell

- FILL view: A view of metal fill, which is used for chip finishing and has no logical function, created by the `signoff_metal_fill` command in IC Compiler.
- CONN view: A representation of the power and ground networks of a cell, created by the PrimeRail or IC Compiler tool and used by the PrimeRail tool for IR drop and electromigration analysis.
- ERR view: A graphical view of physical design rule violations found by verification commands in the IC Compiler tool, such as `verify_zrt_route` or `signoff_drc`.

A physical design stored in a Milkyway library must have at least a CEL view, which contains all of the cell information needed for placement, routing, and mask generation. This includes placement information such as tracks, site rows, and placement blockages; routing information such as netlist, pin, route guide, and interconnect modeling information, and all mask layer geometries, which are used for final mask generation.

Each macro cell typically has both a CEL view and a FRAM view. The FRAM view is an abstraction of the cell containing only the information needed for placement and routing, including the cell border, pin locations, via landing areas, and metal blockage areas where routes are not allowed. The process of creating a FRAM view from a CEL view is called blockage, pin, and via (BPV) extraction.

Each gate-level standard cell can also have both a CEL view and a FRAM view. The FRAM view is used for placement and routing, whereas the CEL view is used only for generating the final stream of mask data for chip manufacturing.

Do not attempt to copy, edit, or delete cell view files directly in the Milkyway database using operating system commands such as `cp` or `rm`. Instead, use a compatible tool such as IC Compiler or Milkyway Environment to make any changes to cell views.

In the past, the Milkyway database was used to store logic, timing, and power data files in LM, TIM, and PWR views. However, these types of information are now usually stored in files maintained outside the Milkyway database, such as .db files.

Design and Reference Libraries

The Milkyway database contains the physical library information needed by implementation and analysis tools. The database contains not only leaf-level physical cell information and technology information, but also design-specific physical information such as the placement and routing results. A chip design typically uses multiple libraries, organized as a hierarchy of design libraries and reference libraries.

A Milkyway library that you open for editing is called a design library. You can open one design library at a time. A library can contain any number of cells. You can open and display multiple cells at the same time from one library. A cell can be built using instances of cells from other libraries that are not currently open. These other libraries are called reference libraries.

Feedback

In the Design Compiler, IC Compiler, or Milkyway Environment tool, you can open a Milkyway design library by using the `open_mw_lib` command. In the IC Compiler and Milkyway Environment tools, you can open a cell in that library for editing by using the `open_mw_cel` command.

A design is typically built as a hierarchy of cells. The entire chip is a single cell built out of lower-level blocks, which are also cells. These blocks are built out of smaller blocks, and so on, down to leaf-level cells. The physical representations of these cells are stored in Milkyway libraries.

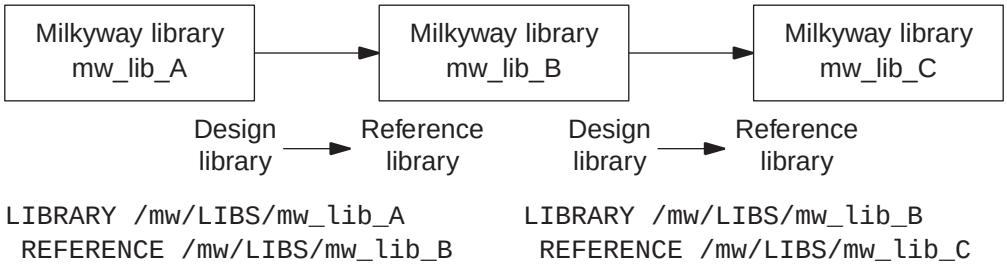
When you open a Milkyway library, you can have both read and write access to the cells in that library. The open library containing the design you are working on is called the design library. The lower-level cells in the design can be found in the same library or in other Milkyway libraries that are not open, called reference libraries. The design you are working on may contain instances of cells that are defined in the reference libraries.

A reference library typically contains data that you do not modify yourself. For example, a reference library might contain standard cells provided by an ASIC vendor, intellectual property provided by an IP block seller, or lower-level blocks designed by another group at your company. Design information that you create or modify yourself must be stored in a design library.

The terms “design library” and “reference library” describe a one-way relationship between two Milkyway libraries. If you define mw_lib_B to be a reference library of design library mw_lib_A, then cells contained in mw_lib_B can be used to build cells contained in mw_lib_A. However, this does not imply that cells contained in mw_lib_A can be used to build cells in mw_lib_B.

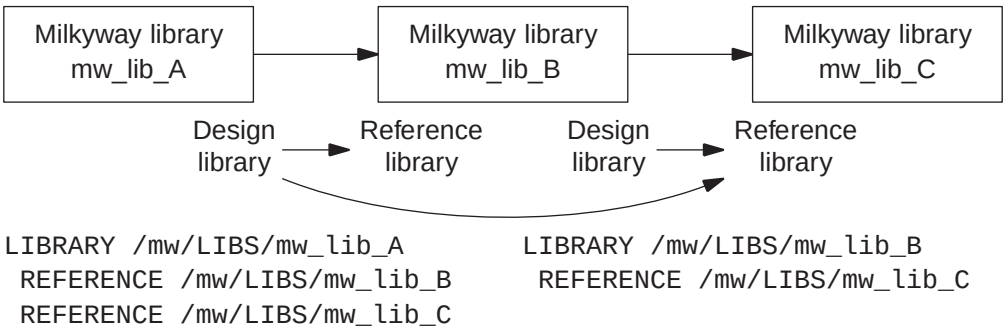
A given Milkyway library can serve as both a design library and a reference library. For example, you can define mw_lib_B to be a reference library of mw_lib_A and define mw_lib_C to be a reference library of mw_lib_B. Then cells in mw_lib_C can be used to build designs in mw_lib_B, and cells in mw_lib_B can be used to build cells in mw_lib_A. Thus, mw_lib_B is a reference library of mw_lib_A and also a design library to mw_lib_C, as shown in Figure 2.

Figure 2 Design and Reference Library Relationships Example 1



Each reference library definition is one level deep. In Figure 2, cells in mw_lib_C cannot be used *directly* in mw_lib_A because the relationship is two levels deep. However, library mw_lib_C can also be explicitly defined as a reference library of mw_lib_A, which would allow its cells to be used directly in mw_lib_A. See Figure 3.

Figure 3 Design and Reference Library Relationships Example 2



The Milkyway library that is currently open is the design library. The associated Milkyway libraries containing read-accessible cells are the reference libraries. You define the association between design and reference libraries with the `set_mw_lib_reference` command.

Setting Milkyway Reference Libraries

In the Design Compiler, IC Compiler, or Milkyway Environment tool, the `set_mw_lib_reference` command defines the Milkyway reference libraries associated with a specified Milkyway design library. To use this command, the Milkyway design library must be *closed*. If the design library is currently open, close it first with the `close_mw_lib` command.

You can specify the list of reference libraries explicitly in the command itself or in a separate text file called the reference control file. You must specify relative or absolute paths to the top-level Milkyway directory names.

For example, to define the Milkyway libraries mw_lib_B and mw_lib_C to be reference libraries of design library mw_lib_A, first close mw_lib_A if it is open. Then use the following command:

```
prompt> set_mw_lib_reference \
          -mw_reference_library {/mw/LIBS/mw_lib_B /mw/LIBS/mw_lib_C} \
          /mw/LIBS/mw_lib_A
```

To define the reference libraries using an external file instead of an explicit list in the command, use the following command:

```
prompt> set_mw_lib_reference \
         -reference_control_file my_refs_A /mw/LIBS/mw_lib_A
```

You specify an absolute or relative path to the reference control file as well as to the Milkyway design library. The file format is described in the next section.

To generate a report showing a list of the reference libraries defined for a particular design library, use the `report_mw_lib` command with the `-mw_reference_library` option. For example,

```
prompt> report_mw_lib -mw_reference_library mw_lib_A
/mw/LIBS/mw_lib_B
/mw/LIBS/mw_lib_C
```

If you do not specify which Milkyway design library to report, the currently open Milkyway library is reported.

Milkyway Reference Control File

You can specify the Milkyway reference libraries associated with a Milkyway design library by using an ASCII text file called a reference control file. The `set_mw_lib_reference` command invokes this file when you use the `-reference_control_file` option.

This is the syntax of the reference control file:

```
LIBRARY path_to_mw_design_library_directory
 REFERENCE path_to_mw_reference_library_directory
 REFERENCE path_to_mw_reference_library_directory
 ...
```

For example,

```
LIBRARY /mw/LIBS/mw_lib_A
 REFERENCE /mw/LIBS/mw_lib_B
 REFERENCE /mw/LIBS/mw_lib_C
```

In this example, mw_lib_B and mw_lib_C are reference libraries of design library mw_lib_A. Cells contained in the reference libraries can be used as instances to build cells contained in the design library.

Specify absolute, not relative, paths to the Milkyway design and reference libraries.

To have the tool generate a reference control file automatically from an existing Milkyway design library that already has reference libraries defined, use the `write_mw_lib_files` command with the `-reference_control_file` option. For example, to generate a

reference control file called my_refs_A for the library mw_lib_A, use the following command:

```
prompt> write_mw_lib_files -reference_control_file \
         -output my_refs_A /mw/LIBS/mw_lib_A
```

This can be done with the mw_lib_A library either open or closed. However, you must specify the absolute or relative path to the design library.

Logic Libraries

The cell logic, timing, and power information is typically contained in a set of Synopsys database (.db) files, which are maintained separately from the Milkyway database. The .db files are produced by the Library Compiler tool from a set of technology characterization files in Liberty (.lib) format.

The Liberty (.lib) files are ASCII-format files that fully describe the cell logic, timing, and power characteristics of the leaf-level logic cells. The Library Compiler tool compiles the .lib files to produce .db files, which contain the same information as the .lib files in a compiled binary format that is more efficient for Galaxy tools to use. In the Design Compiler or IC Compiler tool, you specify the .db files by setting the `search_path`, `target_library`, and `link_library` variables.

The names of the cells in the logical libraries must match the names of the corresponding cells in the physical libraries in the Milkyway database. You can verify that the logical and physical libraries properly match by using the `check_library` command in the Design Compiler, IC Compiler, or Milkyway Environment tool.

Milkyway-Related Commands

Table 3 lists and briefly describes Milkyway-related commands in the IC Compiler and Milkyway Environment tools. These commands let you open, edit, and close Milkyway libraries and cells.

Note:

You must not create, copy, edit, or delete Milkyway database files directly using operating system commands such as `cp` or `rm`. Instead, access the database by using a compatible tool such as IC Compiler or Milkyway Environment and use the tool commands to edit the database contents. This ensures the consistency and integrity of the database.

Feedback

Table 3 Milkyway-Related Commands in the IC Compiler and Milkyway Environment Tools

Command	Action performed
`check_library`	Checks logical and physical libraries for quality and consistency
`close_mw_cel`	Closes one or more open cells in the open Milkyway library
`close_mw_lib`	Closes the current Milkyway library
`convert_mw_lib`	Converts the design data in a Milkyway library to the current schema
`copy_mw_cel`	Copies a cell to create a new cell in the current Milkyway library
`copy_mw_lib`	Copies a Milkyway library
`create_mw_cel`	Creates a new cell in the current Milkyway library
`create_mw_lib`	Creates a new Milkyway library
`current_mw_cel`	Specifies or reports the current cell on which commands operate
`current_mw_lib`	Reports the current Milkyway library; creates a collection containing that library
`extend_mw_layers`	Extends the number of layers supported by the tool
`get_mw_cels`	Creates a collection of cells in the current Milkyway library that meet specified criteria
`list_mw_cels`	Lists the cells contained in the current Milkyway library
`move_mw_cel_origin`	Moves the origin of a Milkyway design
`open_mw_cel`	Opens a cell in the current Milkyway library for viewing or editing
`open_mw_lib`	Opens a Milkyway library for editing, making it the current Milkyway library
`rebuild_mw_lib`	Rebuilds a Milkyway library by scanning all cells in the library directory
`remove_mw_cel`	Removes cells from the current Milkyway library
`report_milkyway_version`	Reports the data file version number, Milkyway schema version, and creation information for a cell
`rename_mw_cel`	Renames a cell in the current Milkyway library
`rename_mw_lib`	Renames a Milkyway library
`report_mw_lib`	Reports the unit range or reference libraries of a Milkyway library

Feedback

Table 3 Milkyway-Related Commands in the IC Compiler and Milkyway Environment Tools (Continued)

Command	Action performed
`save_mw_cel`	Saves an edited cell into the current Milkyway library
`set_mw_lib_reference`	Sets the lower-level reference Milkyway libraries associated with a given Milkyway library
`set_mw_technology_file`	Sets a technology file (.tf, .plib, or .alf) associated with the current Milkyway library
`set_stream_layer_map_file`	Saves a stream-in or stream-out layer mapping file into a Milkyway library
`split_mw_lib`	Copies a specified design and its subdesigns in a Milkyway library into new, separate Milkyway libraries
`write_mw_lib_files`	Writes the library technology information or library reference information to a file

Table 4 lists and briefly describes Milkyway-related commands used in the Design Compiler tool. These commands let you open, edit, and close Milkyway libraries, but not individual cells. The Design Compiler tool lets you open a Milkyway library with the `open_mw_lib` command, write the current design into a Milkyway cell with the `write_milkyway` command, and close the library with the `close_mw_lib` command.

Table 4 Milkyway-Related Commands in the Design Compiler Tool

Design Compiler command	Action performed
`check_library`	Checks logical and physical libraries for quality and consistency
`close_mw_lib`	Closes the current Milkyway library
`copy_mw_lib`	Copies a Milkyway library
`create_mw_lib`	Creates a new Milkyway library
`current_mw_lib`	Reports the current Milkyway library; creates a collection containing that library
`extend_mw_layers`	Extends the number of layers supported by the tool
`open_mw_lib`	Opens a Milkyway library for editing, making it the current Milkyway library
`rebuild_mw_lib`	Rebuilds a Milkyway library by scanning all cells in the library directory

Feedback

Table 4 Milkyway-Related Commands in the Design Compiler Tool (Continued)

Design Compiler command	Action performed
`rename_mw_lib`	Renames a Milkyway library
`report_mw_lib`	Reports the unit range or reference libraries of a Milkyway library
`set_mw_lib_reference`	Sets the lower-level reference Milkyway libraries associated with a given Milkyway library
`set_mw_technology_file`	Sets a technology file (.tf, .plib, or .alf) associated with the current Milkyway library
`write_milkyway`	Writes the design information into a cell in the Milkyway database, including the floorplan, placement, and routing data, if any; the `mw_design_library` variable must be set first
`write_mw_lib_files`	Writes the library technology information or library reference information to a file

Physical Library Data Preparation

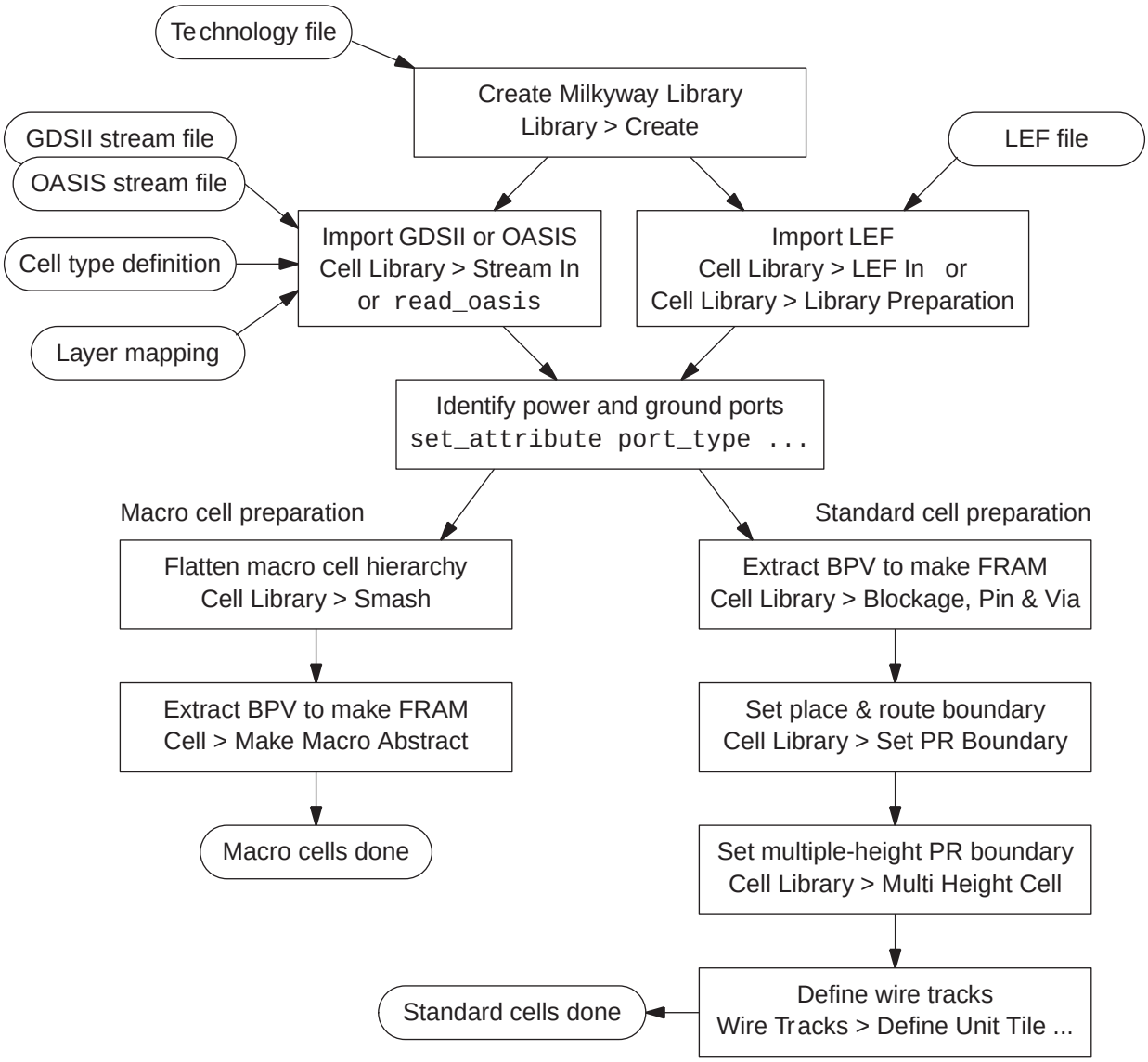
An important task in the IC design flow is the preparation of standard cells and macro cells from physical data obtained from an external source. This information is usually provided in one of the following standard data interchange formats:

- GDSII stream format, a well-established industry-standard data exchange format for integrated circuit layout information
- OASIS, a newer data stream format designed as an improved replacement for GDSII
- LEF/DEF, a set of standard data exchange formats for physical libraries (Library Exchange Format) and design data (Design Exchange Format)

Figure 4 summarizes the library cell preparation steps using the Milkyway Environment tool. You start by creating a new Milkyway library. Then you import the physical data in GDSII, OASIS, or LEF/DEF format and provide any additional information needed to create Milkyway CEL models for the cells in the library. You then specify the power and ground ports, create the FRAM views of the cells, and specify other physical properties required in the FRAM model. The specific steps you need to perform depend on whether you are preparing macro cells or standard cells.

Feedback

Figure 4 Library Preparation Flow in the Milkyway Environment Tool



For detailed information about the library data preparation flow in the Milkyway Environment tool, see the *Library Data Preparation for IC Compiler User Guide*, available on SolvNet in the IC Compiler documentation set.

Feedback

Database Naming Conventions

Objects in the Milkyway database must follow certain naming conventions. If you plan to use the database with other tools or export data to other formats, you need to also follow the conventions of those tools or formats.

Character Restrictions

For most database objects, names can contain any characters, but for libraries and cells, names are restricted to the following characters:

- Alphabetic characters (`abc ...`)
- Numeric characters (`0123456789`)
- Underscores (_)
- Dollar signs (`$`)
- Pipe signs (`|`)

Cell names can also contain the following special-function characters, which are added by the Synopsys tool:

- Periods (`.`) separate cell names from their extensions, which indicate the type of cell.
- Semicolons (`;`) separate cell names, including extensions, from their version numbers. For example, `cellA.cel;2` indicates version 2 of `cellA.cel`.

Name Length Limitations

Table 5 shows the maximum allowed lengths of Milkyway database object names.

Table 5 Name Length Limitations for Milkyway Database Objects

Object type	Name length limitation
application	31 characters
cell	128 characters
cell instance	1,023 characters
color (defined in the technology file)	31 characters
contact (defined in the technology file)	31 characters
error class	15 characters

Feedback

Table 5 Name Length Limitations for Milkyway Database Objects (Continued)

Object type	Name length limitation
group	31 characters
layer	31 characters
library	255 characters
line style (defined in the technology file)	15 characters
mask layer (defined in the technology file)	31 characters
net	1,023 characters
pad	31 characters
pin	31 characters
port	1,023 characters
property	35 characters
reference library	127 characters
region pin	31 characters
stipple pattern (defined in the technology file)	15 characters
technology file	31 characters
tile array class (defined in the technology file)	31 characters
tile array class subtile (defined in the technology file)	31 characters
units (defined in the technology file)	31 characters