



# Database Library Migration Tools

---

## Summary

Application Note  
AP0143 (v1.0) June 20, 2006

This application note provides detailed information on the migration tools associated with Altium Designer's Database Library features (DBLib and SVNDBLib). Direct support for OrCAD CIS is also covered.


---


Altium Designer provides the ability to place components directly from a company database by creating and using a database library. The type of database library used will depend on how you wish to handle your source symbol and model libraries. If the libraries are to be kept in a location on a hard disk or network drive, you would simply use a Database Library (DBLib). If, on the other hand, you wish to place your libraries under source control – using a Subversion repository – you would use an SVN Database Library (SVNDBLib).


Regardless of the type of database library used, the underlying principal of the feature remains the same in each case – the ability to place directly from the linked external database. To make this powerful feature as accessible as possible, tools are provided that enable you to quickly move existing libraries into the database library structure. These tools allow you to migrate from:


- An Integrated Library to a Database Library (DBLib or SVNDBLib)
- A Database Library (DBLib or SVNDBLib) to an Integrated Library
- Source Schematic/PCB libraries to an SVN Database Library
- A Database Library to an SVN Database Library
- An OrCAD Component Information System (CIS) to a Database Library (DBLib).

The following sections of this document take a closer look at how these migrations are performed within Altium Designer.

 For background information on components and libraries, refer to the [Component, Model and Library Concepts](#) article.

 For more information on integrated libraries, refer to the [Enhanced Library Management Using Integrated Libraries](#) article.

 For more information on setting up and using a Database Library, refer to the [Using Components Directly from Your Company Database](#) application note.

 For more information on using an SVN Database Library, refer to the [Working with Version-Controlled Database Libraries](#) application note.

# Creating a Database Library from an Integrated Library

Integrated libraries are, by nature, inherently secure. Added to this is their ideal portability for designs that leave you company site. If the design is to be kept on-site and/or you want to have your Altium Designer components tightly coupled to your company database, then Altium Designer's Database Libraries are the perfect choice. Altium Designer provides the ability to quickly convert your company Integrated libraries into the Database Library (DBLib) or SVN Database Library (SVNDBLib) structure. Multiple integrated libraries may be included in the conversion, with each one being added as a separate table to the target database.

## Converting to a DBLib

With a Database Library file (\*.DBLib) active, simply choose **Tools » Import from Integrated Libraries** from the main menus to access the *Integrated Library to Database Library Translator Wizard*.

The Wizard essentially decompiles nominated integrated libraries, with each library used to build a separate database table in a chosen target database, complete with parameter and model information extracted from the components therein. A specified Database Library file is then used to provide connection to that database.

The Wizard will only extract footprint model information – in terms of model reference and path to that model. For PCB3D and Simulation models, link information will need to be entered manually into the external database.

## Specifying the Target Database

Use the initial page of the Wizard (Figure 1) to specify the target database – either a newly created Microsoft Access 2000 database, or an existing one. For an existing database, if a table already exists with the same name as the integrated library, the information from that library will be appended to the existing table.

**Note:** If creating a new database, click on the folder symbol – to the right of the **Database Location** field – to access a standard *Open* dialog. Use this dialog to determine where, and under what name, the new database is to be created. The chosen name/path will be entered into the **Database Location** field.

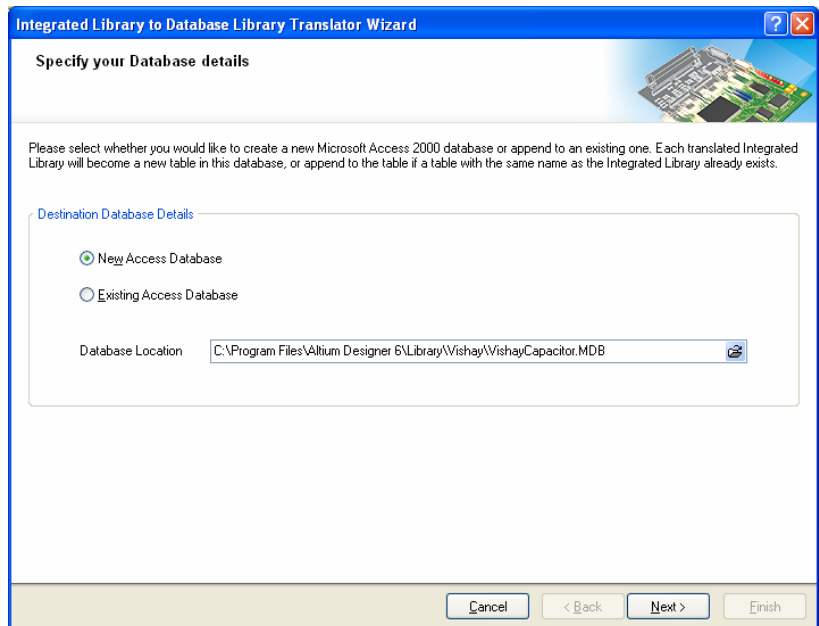


Figure 1. The Integrated Library to Database Library Translator Wizard.

## Specifying the Target Database Library

The second page of the Wizard allows you to specify the target Database Library file.



Figure 2. Specifying the target Database Library.

Either specify the path and name for a new Database Library file to be created or browse to and open an existing file. Typically, you would use an existing DBLib file when converting one or more integrated libraries into the existing Access database to which the DBLib file is currently connected. If you do use an existing DBLib file and the target database is changed, after the Wizard finishes, the DBLib file will be connected to the new target database.

## Choosing the Integrated Libraries

Use the third page of the Wizard to specify the integrated libraries that you wish to convert. Use the **Add** button to access the *Select Source Integrated Libraries* dialog, from where you can browse to and select the required libraries. The constituent schematic symbol and model libraries (where they exist) will be extracted and saved into the location specified in the **Destination Folder** field.

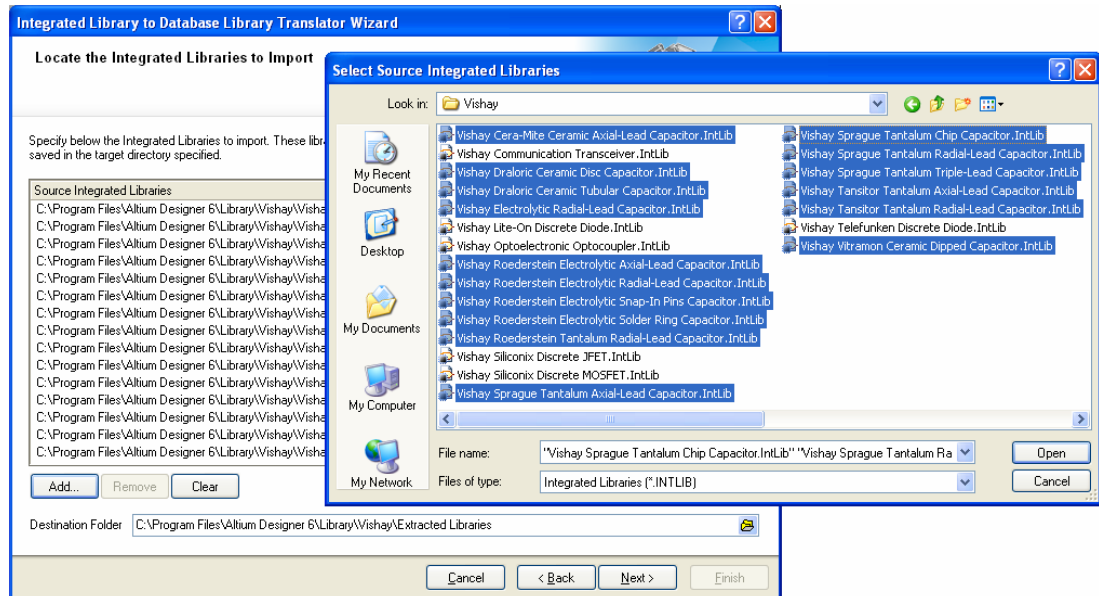


Figure 3. Selecting the integrated libraries to be translated.

### Proceeding with the Conversion

After choosing the source integrated libraries, click **Next** to proceed with the conversion. A progress bar will be displayed, along with information on the current library being translated. After the conversion has completed, click **Finish** to make the specified Database Library file active in the main design window (Figure 4).

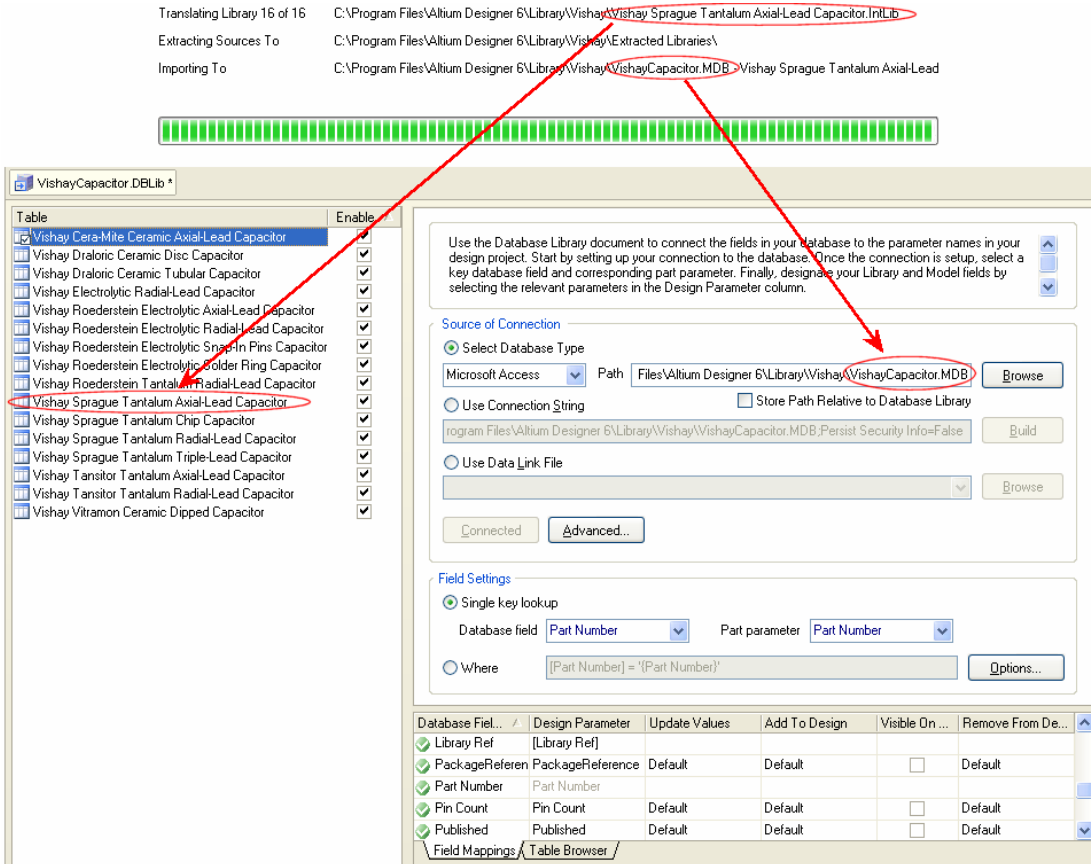


Figure 4. Resulting DBLib document after the translation process is completed.

With the translation process complete, you can then go into the source schematic libraries and remove all parameter and model information from the symbols. Then configure the DBLib document to reference the appropriate database columns.

For more information, refer to the *Mapping Database Fields to Design Parameters* section of the *Using Components Directly from Your Company Database* application note.

## Converting to an SVNDBLib

Conversion to an SVN Database Library from an integrated library is performed using the *SVN Database Library Conversion Wizard* (Figure 5). This Wizard can be accessed by:

- Right-clicking on the entry for a library in the **Projects** panel, and choosing the **SVN Database Library Maker** command.
- Choosing the **SVN Database Library Maker** command from the main **Tools** menu in either the Schematic or PCB Library Editors.

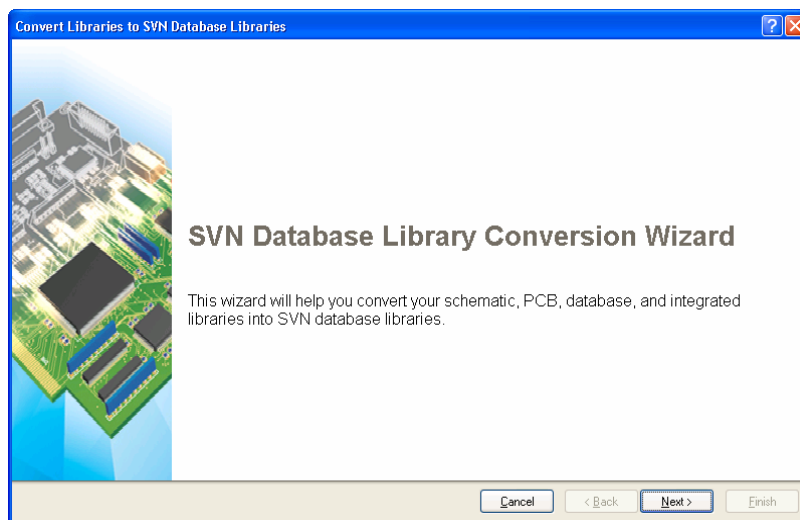


Figure 5. Running the Wizard used to convert libraries to SVN Database Libraries.

## Choosing the Integrated Libraries

Use the initial page of the Wizard to specify the integrated libraries that you wish to convert. Ensure that the **Schematic, PCB, and Integrated Libraries** option is selected.

The **Libraries to Convert** list will initially be pre-populated with one or more libraries. The exact libraries will vary and will depend on how the Wizard was accessed. Typically, these will be schematic and/or PCB libraries. Simply remove these from the list before choosing the integrated libraries you wish to convert.

Use the **Add** button to access the *Library Files* dialog, from where you can browse to and select the required integrated libraries (Figure 6).

## Database Library Migration Tools

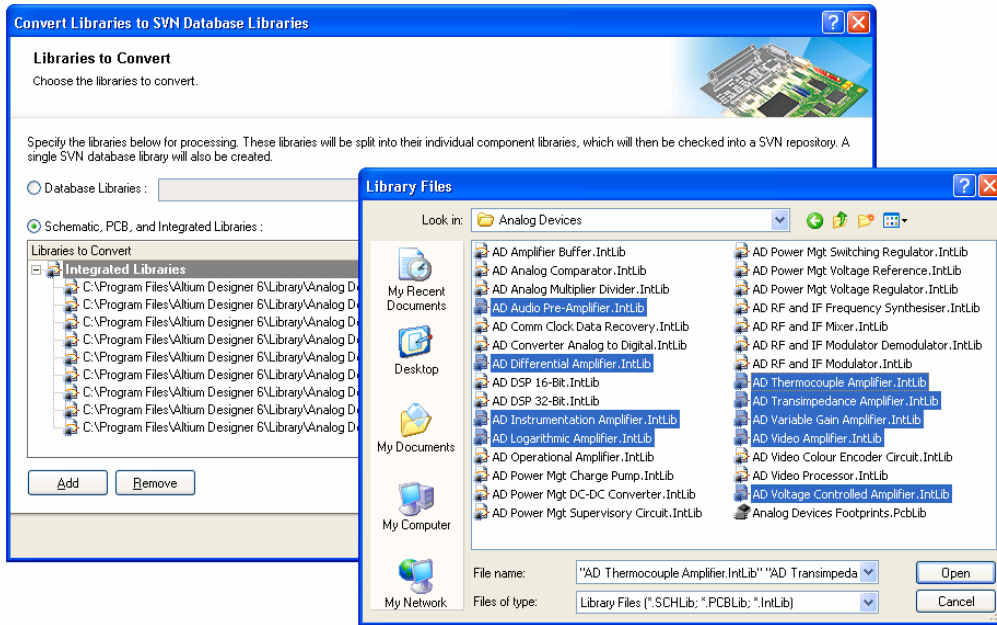


Figure 6. Selecting the integrated libraries to be converted.

## Specifying Conversion Options

Use the second page of the Wizard (Figure 7) to define the following conversion-specific options:

- The link to the Subversion repository. This involves specifying the method (e.g. File, HTTP) and path to the repository folder, under which the symbol and model libraries will be stored.
- The base directories for symbols and footprints within that repository. The symbol and footprint libraries (one symbol/footprint per library file) will be stored in these nominated directories respectively.
- The path and name of the SVNDBLib file to be created. An Access database will also be created, with the same name and stored in the same location.
- Splitter options to be adhered to when the source schematic and PCB libraries – extracted from the chosen integrated libraries – are split into single symbol/footprint libraries:

For a schematic component library, two options are provided that allow you to strip the parameter and/or model information from each constituent component – leaving just the bare symbol.

An option is also provided to deal with the situation where a specified repository directory already has libraries in it. You can set this option to:

- **Overwrite Existing Files** – with this setting, existing libraries in the repository will be overwritten by newly split libraries of the same name.
- **Append Incrementing Number To File Names** – with this setting, the Wizard will scan through the target repository directory to build a list of existing files to be protected. Each potential split library with the same name as an existing library will be appended with the suffix `_n` (where `n` is an integer, starting from 1).

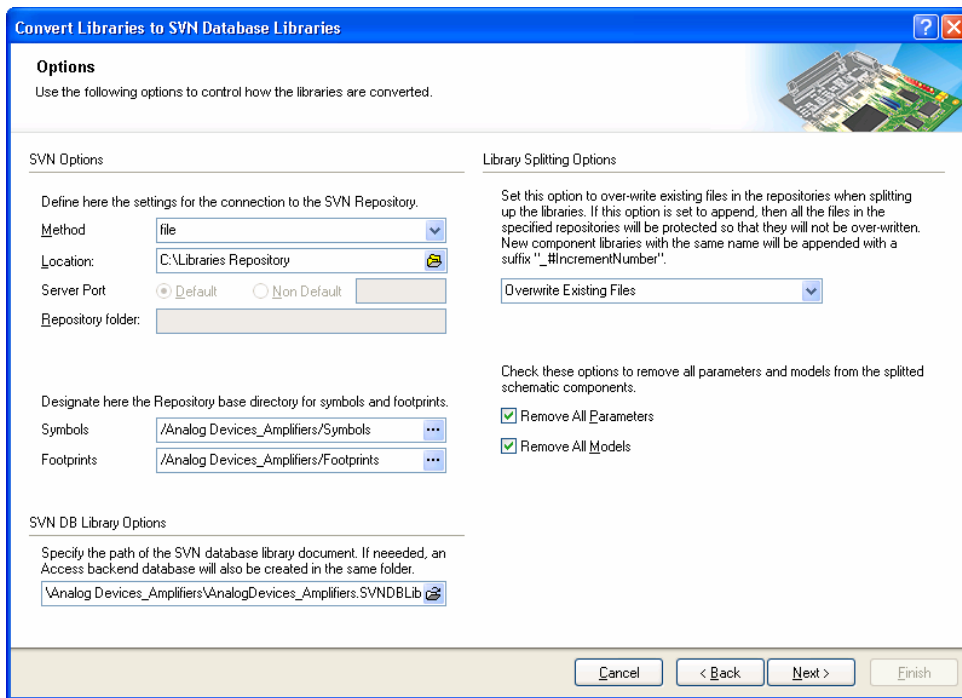


Figure 7. Defining options for the conversion process.

## Proceeding with the Conversion

After choosing the source integrated libraries and setting the related conversion options as required, click **Next** to proceed with the conversion. A progress bar will be displayed, along with information on the current library being converted. The conversion process involves:

- Extraction of the source libraries from the supplied integrated libraries
- Splitting the extracted schematic and PCB libraries into single symbol/footprint library files
- Committing the split symbol and footprint libraries to the repository, in the specified base directories.
- Building a separate database table in the generated Access database, for each integrated library being converted, complete with parameter and model information extracted from the components therein. Each table is named using the name of the integrated library, with an `_IntLib` suffix (e.g. `AD Differential Amplifier_IntLib`).
- Creating the specified SVNDBLib file, connecting to the database and repository.

The Wizard will only extract footprint model information – in terms of model reference. Linked PCB3D and Simulation models are not currently supported for an SVNDBLib. Where such links exist, they will be added as parameters.

After the conversion has completed, click **Finish** to make the SVNDBLib file active in the main design window. Figure 8 illustrates a resulting SVNDBLib file and highlights the connection to the external database (with the same name) as well as the link to the SVN repository and location of the symbol and footprint libraries.

## Database Library Migration Tools

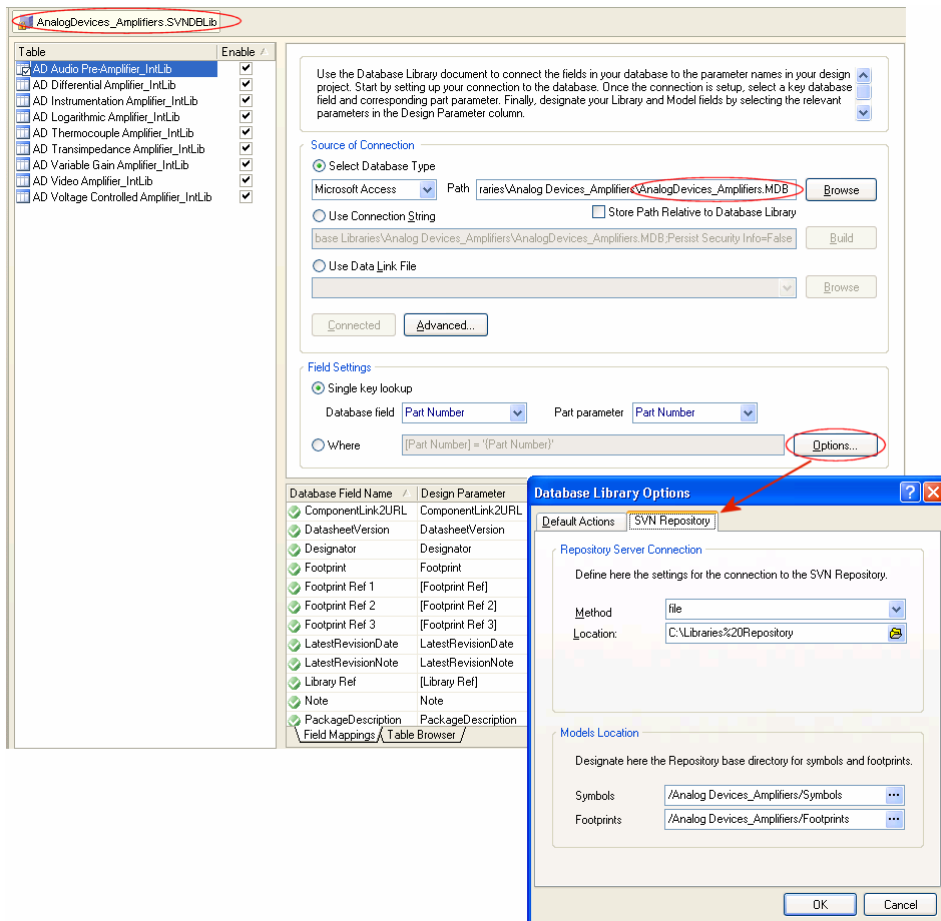



Figure 8. Generated SVNDBLib file linked to repository and database.

 For more information on using your newly-created SVN Database Library, refer to the [Working with Version-Controlled Database Libraries](#) application note.

## Creating an Integrated Library from a Database Library

Altium Designer's Database Libraries are an ideal choice if you want your Altium Designer components to be tightly coupled to your company database. If the design needs to leave your company site, or if you prefer to have your designers work from secure integrated libraries, this can be readily achieved. Altium Designer provides the facility to compile an integrated library directly from a database library – either a non-version-controlled Database Library (DBLib), or a version-controlled SVN Database Library (SVNDBLib). In this way, your CAD Librarians can still use database/version-controlled libraries, while your designers use regularly regenerated integrated libraries, working in an 'offline' fashion as it were.

The process of conversion to an integrated library is carried out on a per-database-table basis. You have full control over which tables in the database – linked to your database library – are considered in this process. A separate integrated library will be generated for each included table.

### Setting up the Conversion

Conversion is performed using the *Offline Integrated Library Maker Wizard* (Figure 9). This Wizard is accessed from either the active DBLib or SVNDBLib document using the **Tools » Offline Integrated Library Maker** command.

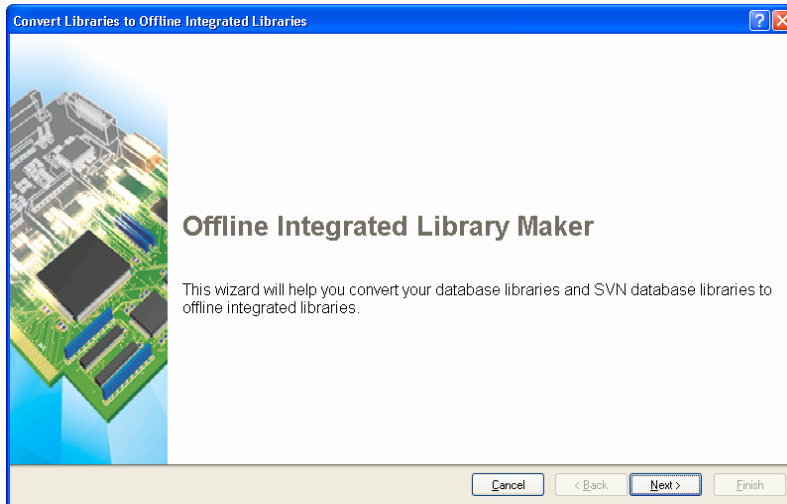


Figure 9. Running the Wizard used to convert database libraries to 'offline' integrated libraries.

### Choosing the Database Library to Convert

Use the initial page of the Wizard to specify which database library (DBLib or SVNDBLib) is to be converted. The active library from which the Wizard was accessed will be specified by default. Should you wish to choose a different database library, simply click on the folder icon to the right of the field. From the *Library Files* dialog that appears, you can browse to and select the required DBLib or SVNDBLib (Figure 10).

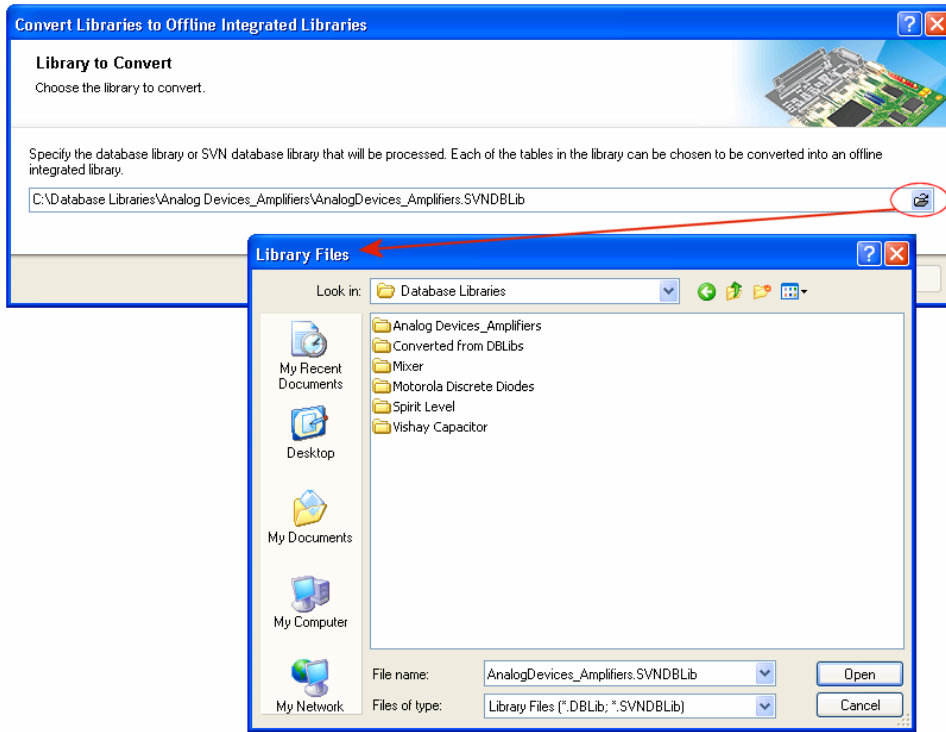


Figure 10. Selecting the database library to be converted.

### Selecting which Tables to Convert

The second page of the Wizard allows you to control which of the tables in the linked database are to be included in the conversion. A separate integrated library will be created for each database table that you include.

All database tables are included in the conversion by default. To exclude a table, simply ensure that its associated **Convert** option is disabled.

This page of the Wizard also enables you to nominate an output directory, in which the generated integrated libraries will be stored (Figure 12). Enter the path to this base directory directly, or click on the folder icon to the right of the field to access the *Browse for Folder* dialog, from where you can browse to and select the required directory. The generated output for each included database table will be stored in its own sub-folder within this base directory, named using the table's name.

Database Tables	Convert
<input type="checkbox"/> AD Audio Pre-Amplifier_IntLib	<input checked="" type="checkbox"/>
<input type="checkbox"/> AD Differential Amplifier_IntLib	<input checked="" type="checkbox"/>
<input type="checkbox"/> AD Instrumentation Amplifier_IntLib	<input checked="" type="checkbox"/>
<input type="checkbox"/> AD Logarithmic Amplifier_IntLib	<input checked="" type="checkbox"/>
<input type="checkbox"/> AD Thermocouple Amplifier_IntLib	<input checked="" type="checkbox"/>
<input type="checkbox"/> AD Transimpedance Amplifier_IntLib	<input checked="" type="checkbox"/>
<input type="checkbox"/> AD Variable Gain Amplifier_IntLib	<input checked="" type="checkbox"/>
<input type="checkbox"/> AD Video Amplifier_IntLib	<input checked="" type="checkbox"/>
<input type="checkbox"/> AD Voltage Controlled Amplifier_IntLib	<input checked="" type="checkbox"/>

Figure 11. Decide which of the tables in the linked database will be included in the conversion.

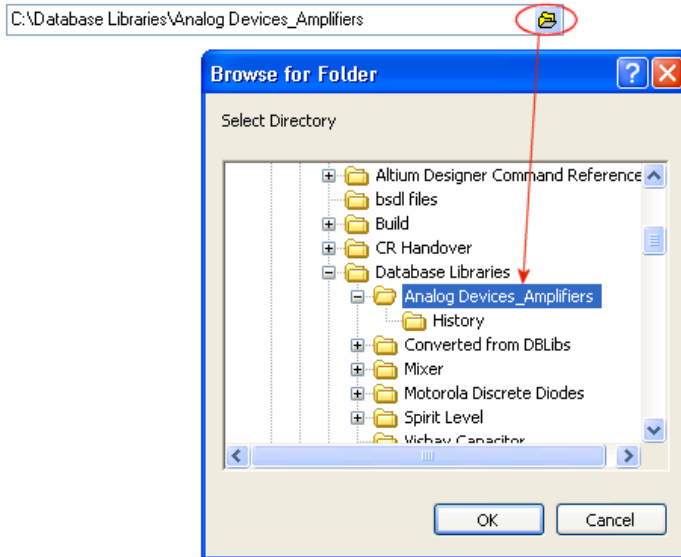


Figure 12. Specify a base directory for all generated output.

## Proceeding with the Conversion

After choosing the database library and setting the related conversion options as required, click **Next** to proceed with the conversion. A progress bar will be displayed, along with information on the current database table being converted. Remember that the conversion process is carried out for each database table you have nominated to convert. The following is essentially a breakdown of this process:

- An integrated library package (\*.LibPkg) is created and opened in the **Projects** panel. The package is named using the name of the table. For a table named AD Differential Amplifier\_IntLib for example, this would give AD Differential Amplifier\_IntLib.LibPkg.
- A schematic library document is created and added to the LIBPKG. The schematic is again named using the table's name (e.g. AD Differential Amplifier\_IntLib.SchLib).
- Each record in the table is then considered and the appropriate Altium Designer component built. To do this, the referenced schematic symbol is retrieved from the appropriate source library and added as a component to the new schematic library document. Parameter and model link information defined in the record is then added to that component.
- The referenced footprint models for the record are retrieved and added to a PCB library document. This document is again named after the table (e.g. AD Differential Amplifier\_IntLib.PcbLib). The PCB library document is then added to the LIBPKG.
- If the source library is a DBLib and PCB3D and/or Simulation model links have been defined in the database record, the referenced PCB3D library and Simulation model files are also added to the LIBPKG. The location of such models remains unchanged. The full path to a model is specified as part of its corresponding model link.

## Database Library Migration Tools

- The LIBPKG is then compiled to give the integrated library (e.g. AD Differential Amplifier\_IntLib.IntLib), which is subsequently added to Altium Designer's Installed Libraries.

Figure 13 shows an example of initial tables listed in a database library, the creation of source library package projects for each of those tables, and the resulting addition of the subsequently-compiled integrated libraries to the Installed Libraries list.

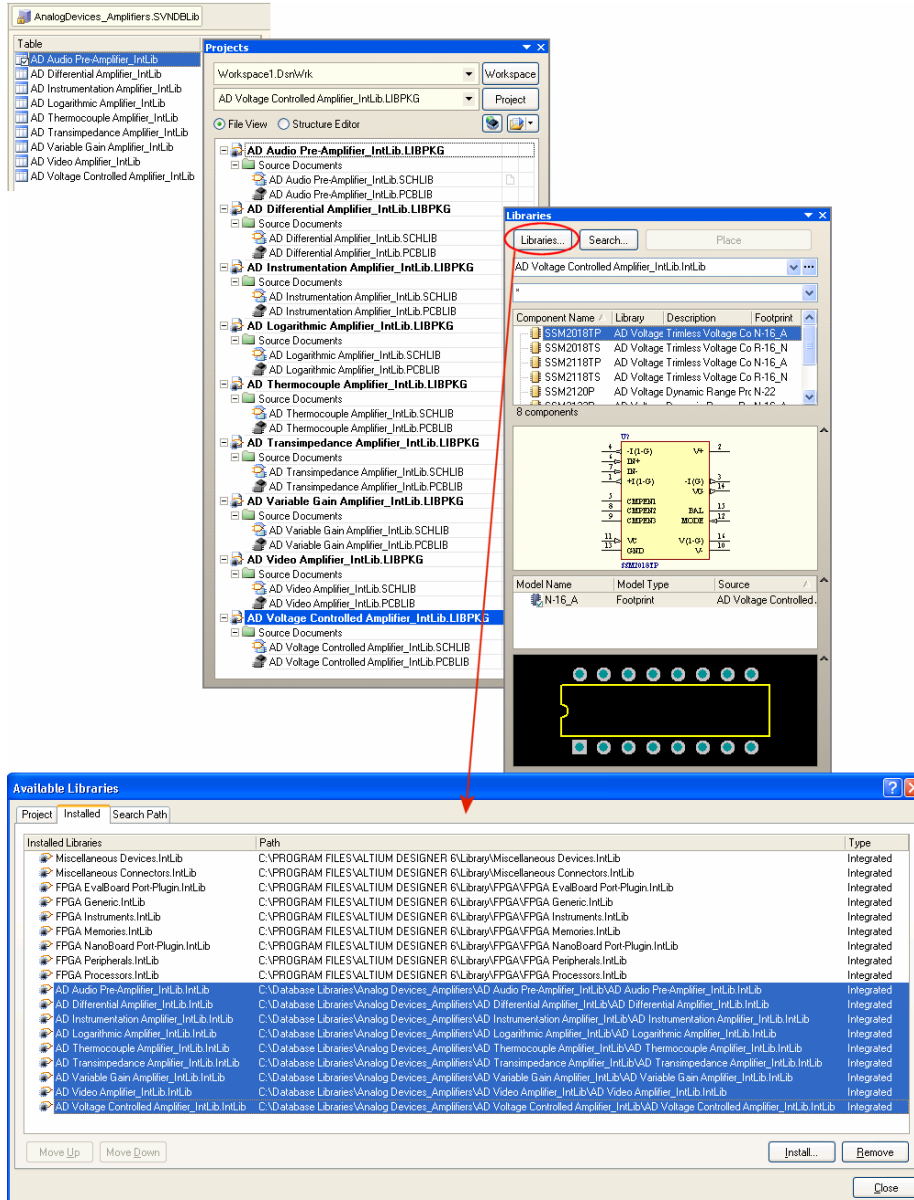


Figure 13. Resulting integrated libraries are added to the Installed Libraries.

All documents – LIBPKG, SCHLIB, PCBLIB and INTLIB – are stored in a sub-folder in the output directory nominated in the Wizard. Once again, the table name is used to name this sub-folder.

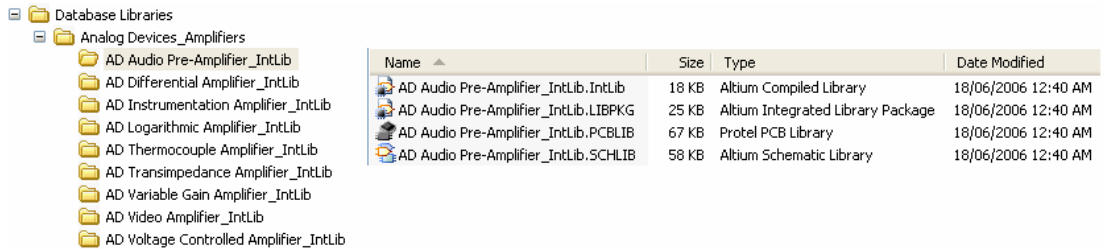




Figure 14. Storage of generated output – one sub-folder per database table.

-  For more information on setting up and using a Database Library, refer to the [Using Components Directly from Your Company Database](#) application note.
-  For more information on using an SVN Database Library, refer to the [Working with Version-Controlled Database Libraries](#) application note.

## Creating an SVNDBLib from Source Libraries (Sch/Pcb)

---

You may have opted to use source schematic and PCB libraries – added to the design project – rather than compiling them into an integrated library. Maybe you wish to edit the source components in these libraries frequently, and prefer not to decompile and recompile the corresponding IntLib each time. Version-controlled database libraries offer a similar deal – the ability to quickly access the source libraries for modification, coupled with the security of having those libraries stored in a source control repository.

Altium Designer facilitates the quick and simple conversion from your existing source schematic and PCB libraries, into the SVN Database Library structure. Conversion is performed with the *SVN Database Library Conversion Wizard*, in much the same way as when converting an integrated library. Multiple libraries of each type may be included in the conversion, with each schematic library added as a separate table to the target database.

### Choosing the Source Libraries

The source libraries to be used in the conversion are specified on the initial page of the Wizard. As the Wizard pre-populates with library entries based on where it is accessed from, you can save time by launching the Wizard from the right place to begin with:

- **From the Schematic Library Editor** – with the source schematic library open, access to the Wizard can be made using the Tools » SVN Database Library Maker command. However, should you wish to involve the linked footprint models in the conversion, you will need to browse for, and add, the respective PCB footprint libraries.
- **From the PCB Library Editor** – with the source PCB library open, access to the Wizard can be made using the Tools » SVN Database Library Maker command. However, conversion of the PCB footprint libraries alone is not a typical scenario and you will therefore need to browse for, and add, the respective schematic component libraries.
- **From the Projects panel** – this is by far the easiest method and requires that the source schematic and PCB libraries are added to a project. Quite often, this will be the case, when integrated libraries have not been used. Simply right-click on the entry for a library file and choose the SVN Database Library Maker command. The Wizard will automatically load all project libraries (Figure 15).

Irrespective of how you access the Wizard you can of course modify the list of libraries to be converted, using the **Add** and **Remove** buttons. The former gives access to the *Library Files* dialog, from where you can browse to and select schematic and PCB libraries to add to the list.

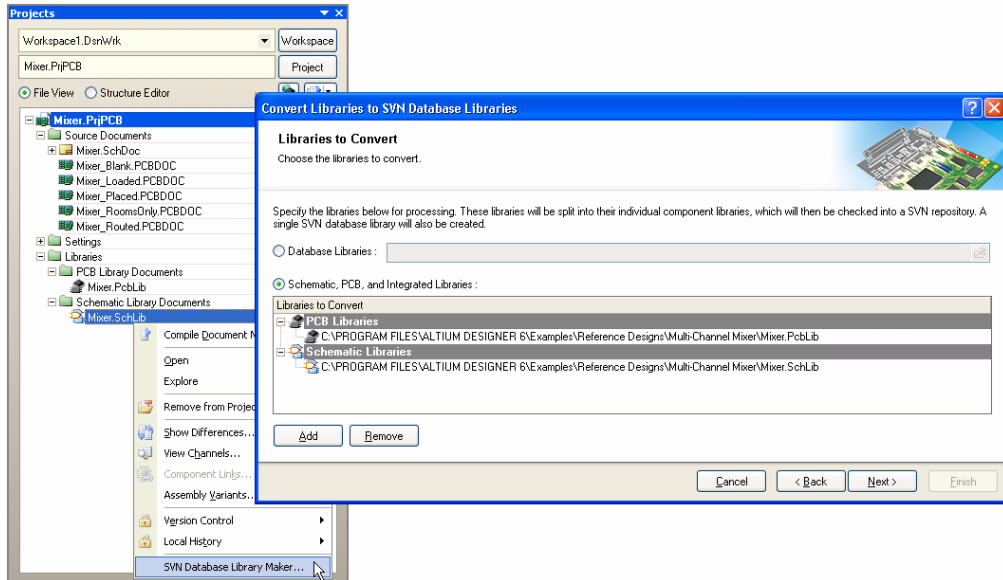


Figure 15. Populating the list of libraries to be converted, automatically, with all source libraries in a project.

## Conversion Options

Use the second page of the Wizard to define conversion-related options. These are options relating to how the source libraries should be split, in which directories of which repository they should be stored, and the output directory for the SVNDBLib file. For more information, see the section [Specifying Conversion Options](#), earlier in this document.


## Proceeding with the Conversion

After choosing the source libraries and setting the related conversion options as required, click **Next** to proceed with the conversion. A progress bar will be displayed, along with information on the current library being converted. The conversion process involves:

- Splitting the schematic and PCB libraries into single symbol/footprint library files
- Committing the split symbol and footprint libraries to the repository, in the specified base directories.
- Building a separate database table in the generated Access database, for each schematic library being converted, complete with parameter and model information extracted from the components therein. Each table is named using the name of the schematic library, with an `_SchLib` suffix (e.g. `Mixer_SchLib`).
- Creating the specified SVNDBLib file, connecting to the database and repository.

The Wizard will only extract footprint model reference information. Linked PCB3D and Simulation models are not supported for an SVNDBLib. Where such links exist, they will be added as parameters.

After the conversion has completed, click **Finish** to make the SVNDBLib file active in the main design window.

 For more information on using your newly-created SVN Database Library, refer to the [Working with Version-Controlled Database Libraries](#) application note.

# Converting a DBLib to an SVNDBLib

You may already be enjoying the power of the Database Library feature – having perhaps converted your existing integrated libraries to the DBLib structure, or having created a DBLib from scratch. With the advent of version-controlled database libraries, you may want to move your source symbol and footprint libraries under the protective and secure wing of a source control repository. Altium Designer provides the means to effect this migration, from your current DBLib to an SVNDBLib.

With the original Database Library file (\*.DBLib) open as the active document, run the **SVN Database Library Maker** command from the main **Tools** menu. The *SVN Database Library Conversion Wizard* will appear. The initial page of the Wizard will have the **Database Libraries** option already enabled, with the path to the source DBLib file already entered (Figure 16).

Access to the Wizard can be made from a number of places. You can of course browse for the DBLib you wish to convert, by clicking on the folder icon, to the far right of the **Database Libraries** field.

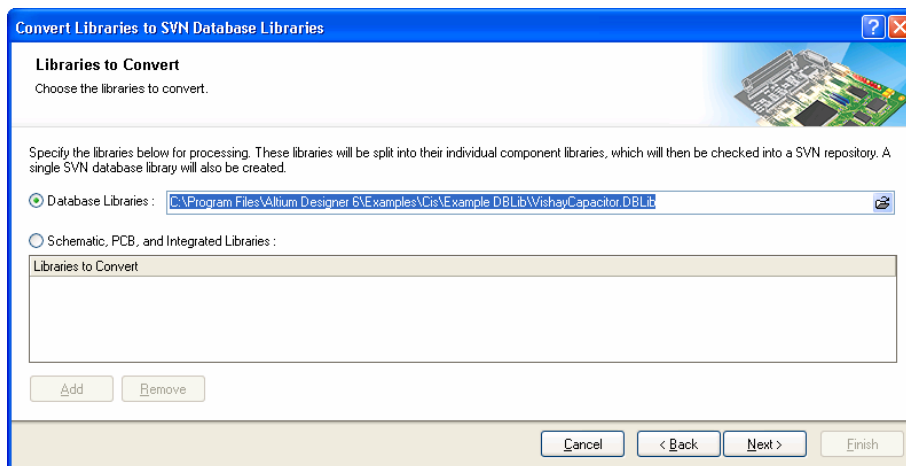


Figure 16. Selecting the Database Library (DBLib) to be converted.

## Conversion Options

Use the second page of the Wizard to define conversion-related options. These are options relating to how the source schematic and PCB libraries should be split, in which directories of which repository they should be stored, and the output directory for the SVNDBLib file. For more information, see the section [Specifying Conversion Options](#), earlier in this document.

**Note:** As the external database already exists – linked to the original DBLib file – one will not be created. The database will remain in its current location. Should you wish to have the generated SVNDBLib file and the database in the same location, you basically have two options:

- Set the output path for the SVNDBLib to be the same directory in which the database currently resides.
- Generate the SVNDBLib in a different directory and then move the database to that directory. You will need to remember to modify the connection within the SVNDBLib file to point to the database in its new location and reconnect.

## Proceeding with the Conversion

After choosing the source DBLib file and setting the related conversion options as required, click **Next** to proceed with the conversion. A progress bar will be displayed, along with information on the current library being converted. The conversion process involves:

- Splitting the schematic and PCB libraries, referenced by the component records in the linked database, into single symbol/footprint library files
- Committing the split symbol and footprint libraries to the repository, in the specified base directories.
- Creating the specified SVNDBLib file, connecting to the database and repository.

After the conversion has completed, click **Finish** to make the SVNDBLib file active in the main design window.

## A Word about Field Mappings...

In the generated SVNDBLib the mappings defined between fields in the database and design parameters, on the **Field Mappings** tab, remain as they were originally defined in the DBLib. This is illustrated by example in Figure 17.

The figure shows two side-by-side screenshots of a software interface. The top window is titled 'Vishay Capacitor.DBLib' and the bottom window is titled 'Vishay Capacitor.SVNDBLib'. Both windows display a 'Table' browser on the left and a 'Field Mappings' table on the right. The 'Field Mappings' table in both windows is identical, showing the following data:

Database Field Name	Design Parameter	Update Values	Add To Design	Visible On Add	Remove From Design
Component Type	Component Type	Default	Default	<input type="checkbox"/>	Default
ComponentLink1Description	ComponentLink1Description	Default	Default	<input type="checkbox"/>	Default
ComponentLink1URL	ComponentLink1URL	Default	Default	<input type="checkbox"/>	Default
ComponentLink2Description	ComponentLink2Description	Default	Default	<input type="checkbox"/>	Default
ComponentLink2URL	ComponentLink2URL	Default	Default	<input type="checkbox"/>	Default
DatasheetDocument	DatasheetDocument	Default	Default	<input type="checkbox"/>	Default
Designator	Designator	Default	Default	<input type="checkbox"/>	Default
Footprint	Footprint	Default	Default	<input type="checkbox"/>	Default
Footprint Path	[Footprint Path]				
Footprint Ref	[Footprint Ref]				
Library Path	[Library Path]				
Library Ref	[Library Ref]				
Note	Note	Default	Default	<input type="checkbox"/>	Default
PackageReference	PackageReference	Default	Default	<input type="checkbox"/>	Default
Part Number	Part Number				
Pin Count	Pin Count	Default	Default	<input type="checkbox"/>	Default
Published	Published	Default	Default	<input type="checkbox"/>	Default
Publisher	Publisher	Default	Default	<input type="checkbox"/>	Default

Figure 17. Field mappings remain the same between the original DBLib and the generated SVNDBLib.

## Database Library Migration Tools

There are, however, two important areas to highlight:

- Any defined path mappings for symbols and footprint models are ignored. The SVNDBLib uses only the mapped reference fields ([Library Ref] and [Footprint Ref]) for locating the required symbol and footprint within the libraries committed to the repository. The path information defined in the database is ignored as it points to libraries located on a hard disk or other local/network medium. This path information can be modified, post conversion, to point to the libraries in the repository. For more information, see the next section – *Modifying Database Path Information*.
- Any defined PCB3D and Simulation model mappings will also remain defined in the SVNDBLib file. However, storage of PCB3D model libraries and simulation model files in the Subversion repository is not supported. Although the model links will be added to a component instance when placed, the models themselves will not be found.

### Modifying Database Path Information

When you place a component from an SVN Database Library its symbol (specified by the [Library Ref] mapping) and footprint model (specified by the [Footprint Ref] mapping) are extracted from symbol and model libraries in the version control repository.

The base directories in which these symbols and models reside are specified as part of the conversion options. Within the generated SVNDBLib file, these directories can be found on the **SVN Repository** tab of the *Database Library Options* dialog.

In the example of Figure 18, the location for the schematic symbols and PCB footprint models have been set to point to the following sub-folders within the repository structure:

- The *SYM* folder for symbols
- The *FTPT* folder for footprints.

It is important to stress that the symbols and footprints must reside within the base repository directories specified. They can of course be in sub-folders of those directories and the paths specified for both symbols and footprints can point to the same directory in the repository.

The following methods can be used to locate the required symbol and footprint model within these nominated base directories:

- **Absolute Path** – the full path to the location of the library can be entered into the database. (e.g. `http://ares/svn/Altium/SVNTest/SchematicSymbols/Capacitor_NonPolarized.SchLib`).
- **Relative Path** – a relative path (relative to the root of the repository) to the location of the library can be entered into the database. The URL for the repository – specified on the **SVN Repository**

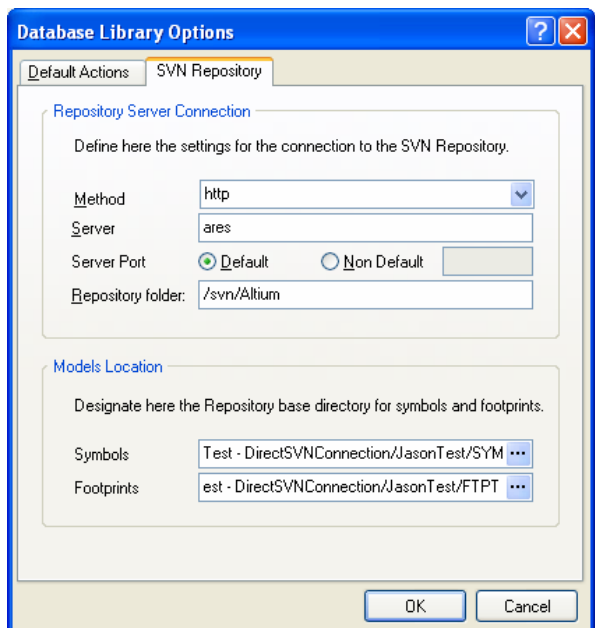


Figure 18. Specifying base repository directories for symbols and footprints.

tab of the *Database Library Options* dialog – will be prefixed to the path you enter, to give the absolute address. Considering Figure 18 for example, if you specify /SVNTest/SchematicSymbols/Capacitor\_NonPolarized.SchLib, the full path will be http://ares/svn/Altium/SVNTest/SchematicSymbols/Capacitor\_NonPolarized.SchLib.

- **Filename Only** – you can simply specify the name of the library in which to find the required symbol or footprint, again within the database record for the component. The first file found with this name will be used. (e.g. Capacitor\_NonPolarized.SchLib).
- **No Path Information** – you can opt not to enter any path information in the database record for the component. When locating the symbol/footprint, the system will initially look for the first library named like the symbol or footprint itself. For example, if the logical symbol name in the database (specified by the [Library Ref] mapping) is DIO-SCHOTTKY-2S, the system will look for the first file named DIO-SCHOTTKY-2S.SchLib and look for the symbol within this file. If the symbol/footprint cannot be found in this way, the system will look for a match in all libraries.


When searching for a symbol/model match, the flattened folder paths in the base symbol or footprint directory are sorted and searched alphabetically.

When the SVNDBLib has been generated from an existing DBLib, there will always be a corresponding library with the name of the actual symbol/footprint.

After conversion from DBLib to SVNDBLib, path information will be ignored, as it points to the original libraries located on a hard disk for example. Should you wish to still include path information in the database, simply modify the information to point to the libraries in the repository – using any of the first three methods listed previously.

It is worth remembering that not specifying library path information in the database makes it much more robust. The repository location and/or its internal folder structure could be changed and the database would not need to be updated.

Database field values can be modified directly from the **Table Browser** tab of the SVNDatabaseLib Editor. For more information, refer to the section *Modifying a Database Table through the SVNDBLib File*, in the [Working with Version-Controlled Database Libraries](#) application note.

 For more information on using your newly-created SVN Database Library, refer to the [Working with Version-Controlled Database Libraries](#) application note.

# Direct OrCAD® CIS Support

Built on the foundation of the database library system, Altium Designer provides full support for connection to, and use of, existing OrCAD Component Information Systems (CIS). The CIS structure is essentially converted into Altium Designer's Database Library structure.

## From OrCAD to Altium Designer – Translations Required

To provide the facility for direct placement from the external database (\*.mdb, \*.xls), the following file translations are required:

- The OrCAD CIS Configuration File (\*.dbc), which handles the link to the external database and includes the database field-to-design parameter mapping information, must be translated into an Altium Designer Database Library file (\*.DBLib).
- The relevant OrCAD library files must be translated into Altium Designer library files:  
OrCAD Capture Library (\*.OLB) → Schematic Library (\*.SchLib)  
OrCAD Max Library File (\*.LLB) → PCB Library (\*.PcbLib).

## Creating the Database Library Automatically

The simplest and most efficient method to create the DBLib file and source Altium Designer libraries, is to use the *Import Wizard* (**File » Import Wizard**). On the second page of the Wizard, ensure that the `OrCAD CIS Configuration Files and Libraries` entry is selected as the import file type (Figure 19).

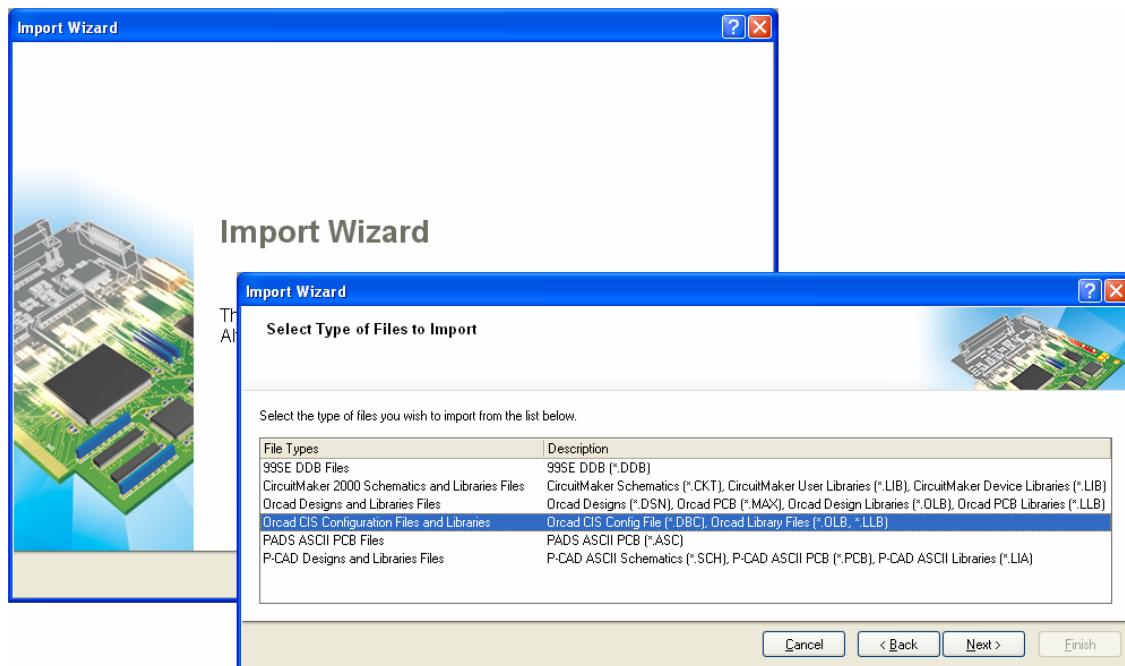


Figure 19. Use the Import Wizard to quickly create the DBLib and source library files.

Follow the subsequent pages of the Wizard, in which you are required to specify:

- The location of the external database
- The location of the CIS Configuration File
- The location of the target DBLib file.
- The OrCAD Schematic and/or PCB libraries referenced by the external database.

When specifying the location of the target DBLib file, either specify the path and name for a new file to be created, or browse to and open an existing file.

When specifying the OrCAD source libraries, you also have control over where the resulting Altium Designer libraries are saved. By default, these libraries will be saved to the sub-folder `Libraries`, located in the same directory as the target DBLib file.

Figure 20 illustrates completed Wizard entries for an example OrCAD database – `BENCH.mdb`.

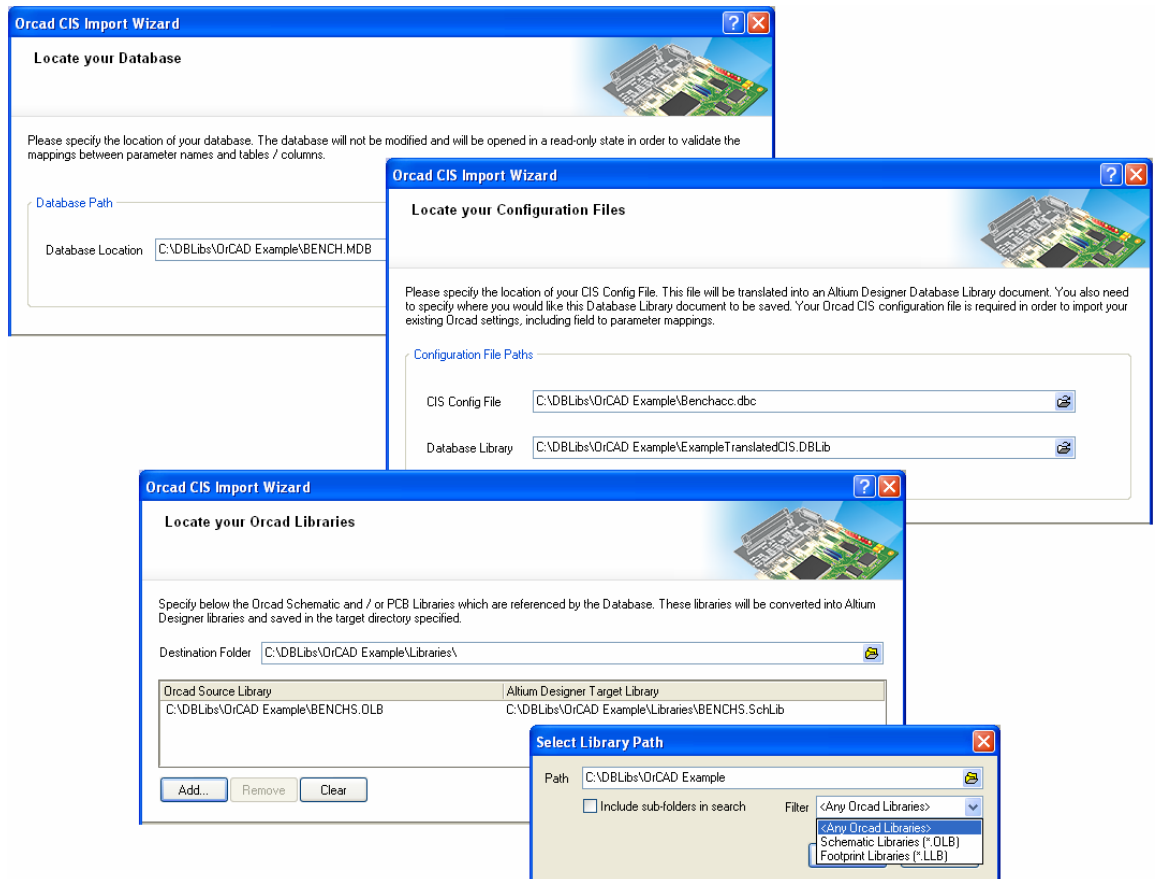


Figure 20. Specify the OrCAD database, CIS Config file and source libraries and the target DBLib file.

### Proceeding with the Import

After specifying the source and target files and directories as required, click **Next** to proceed with the import. After the import is completed, click **Finish** to make the specified Database Library file active in the main design window (Figure 21).

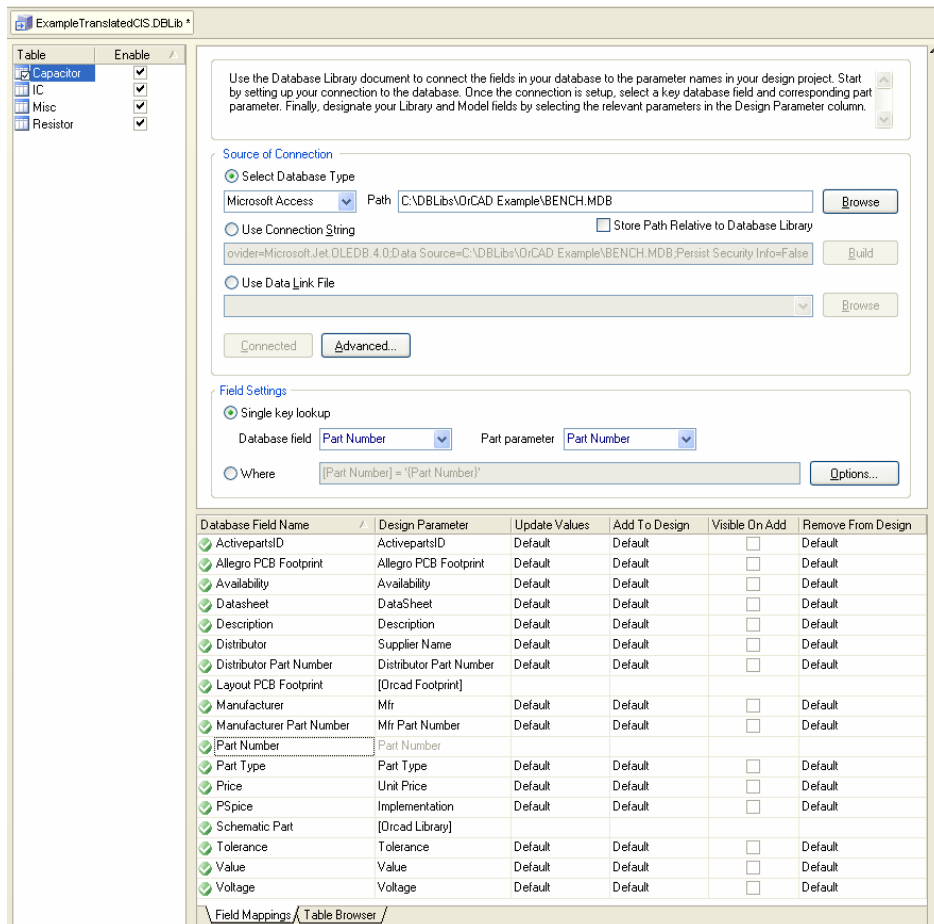


Figure 21. Resulting DBLib document after the OrCAD files have been translated.

The parameter mapping information – including the defined lookup key – is taken directly from the OrCAD CIS file. In addition, you will notice from Figure 21 that the following two model mapping entries are automatically set:

Layout PCB Footprint → [OrCAD Footprint]

Schematic Part → [OrCAD Library]

These entries provide the link to the source schematic symbol for a particular component record in the external database, and the applicable PCB Footprint model linked to that component.


Determine mapped parameter update options as required.

A library search path is automatically added to the DBLib file, which points to the directory containing the translated library files.

## Creating the Database Library Manually

Creation of the DBLib file using the *Import Wizard* is the fastest method, but not the only method. You can of course create the DBLib file manually. First, you will need to create and define the DBLib file:

- Create a new Database Library document (**File » New » Library » Database Library**).
- Setup the connection to the OrCAD external database.
- Once connected, ensure that the field mappings are as required, especially the schematic symbol and footprint model mappings. Define mapped parameter update options as required.

 For more information on setting up the DBLib file, refer to the [Using Components Directly from Your Company Database](#) application note.

You will then need to import the OrCAD libraries. This can be achieved by using the *Import Wizard* (**File » Import Wizard**) to translate the OrCAD library files (\*.OLB, \*.LLB) into Altium Designer libraries (\*.SchLib, \*.PcbLib).

When choosing the file types to import, select the OrCAD Designs and Libraries Files entry. Skip the page for *Importing OrCAD Designs* and proceed to the page for *Importing OrCAD Libraries* (Figure 22). Add all source OrCAD libraries referenced by the database.

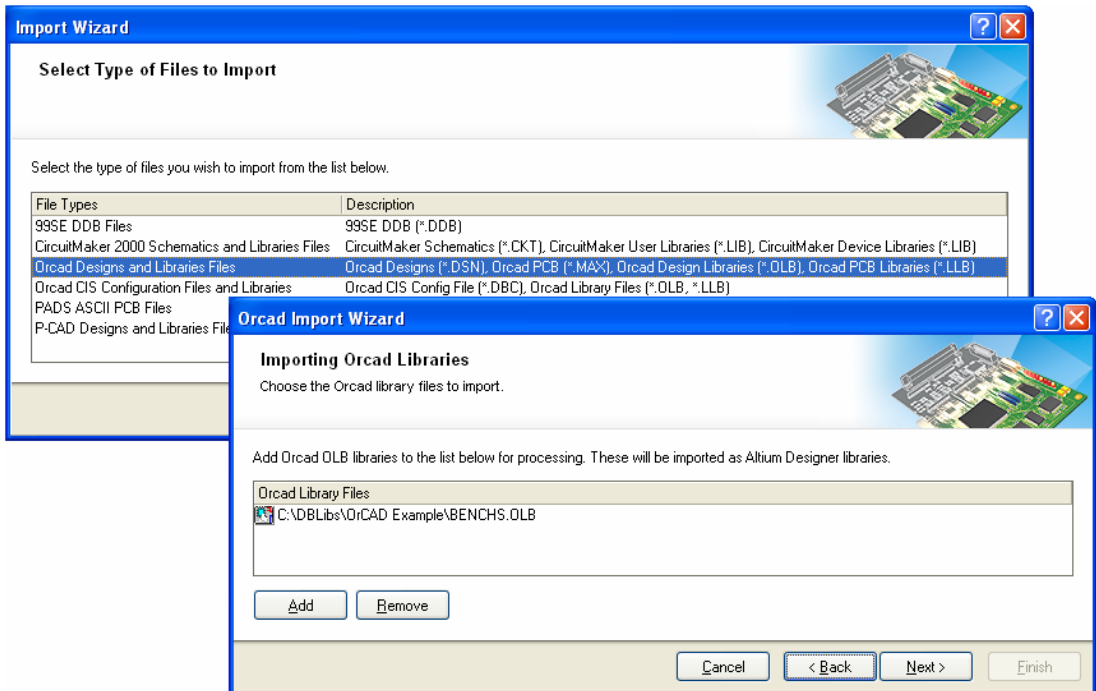


Figure 22. Specify the OrCAD libraries to be translated.

On the *Output Libraries* page of the Wizard (Figure 23), specify the output directory for the generated library files. After the import is complete, a folder – Imported OrCAD Libraries.PrjPcb – will be generated in the nominated directory.

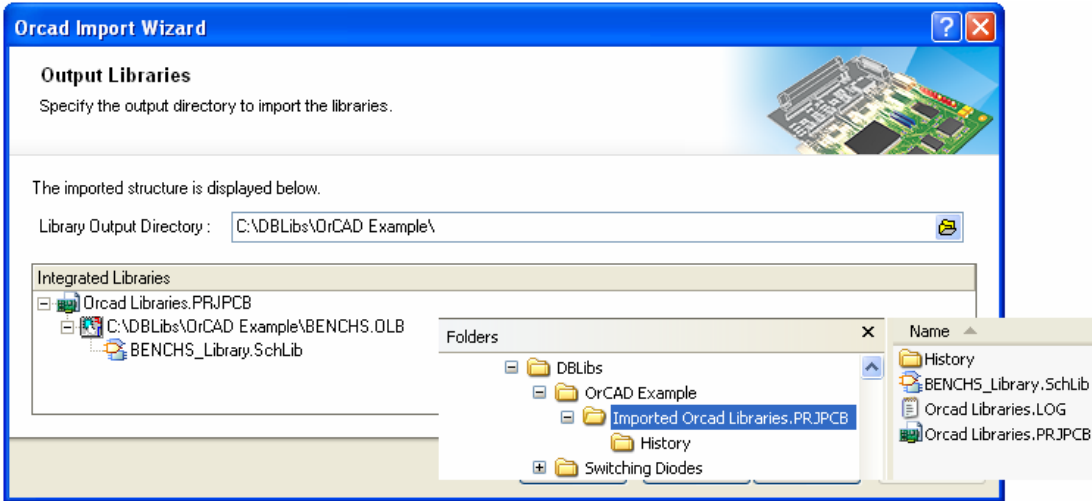


Figure 23. Specify an output folder for the generated Altium Designer libraries.

You can simply move the generated Altium Designer libraries contained within to another location and delete this folder. For example, you may want to move the libraries to a folder named `Libraries` – created within the directory containing the `DBLib` file.

Once you have the symbol and footprint libraries, you will need to go back to the `DBLib` file and set up the library search paths to point to the directory folder in which those libraries are stored.

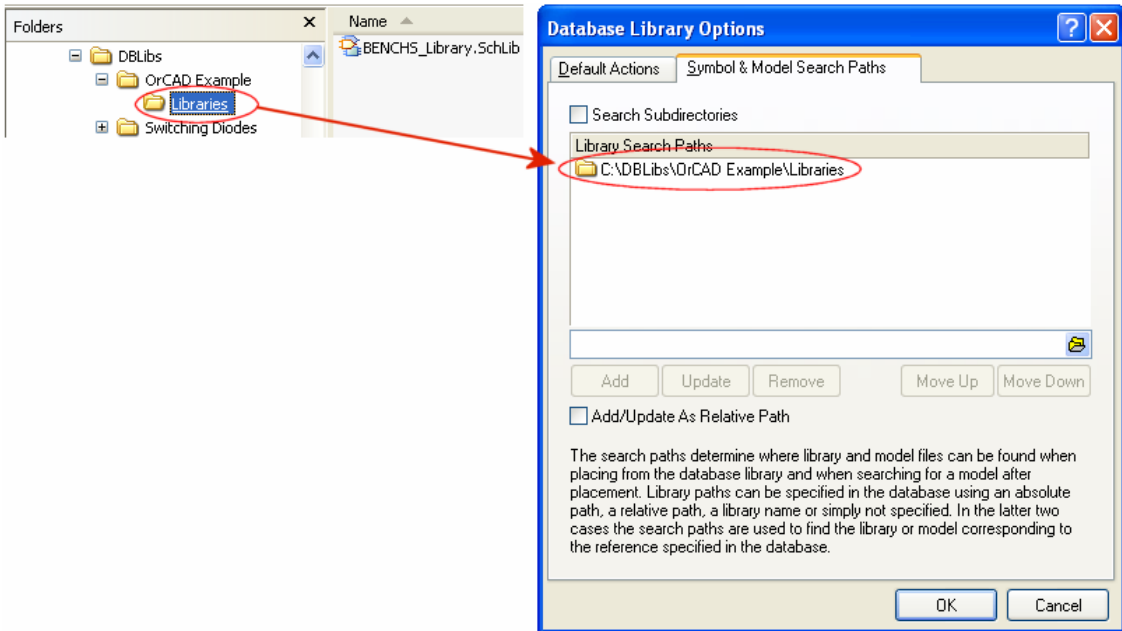


Figure 24. Add a search path to the `DBLib` file, targeting the generated source libraries.

## Placement and Maintenance

Once the DBLib file is created, the mapping defined as required and the source libraries translated and targeted with a suitable DBLib search path, you can add the library to the Available Libraries list. Once added, you can place a component onto the schematic, with the parameter and footprint model information added on-the-fly, directly from the database.



For further information, refer to the section *Countdown to Placement*, in the *Using Components Directly from Your Company Database* application note.

After placement, the chosen key parameter in the DBLib file is used to ensure that the placed component on the schematic retains its link to the corresponding record for that component in the external database. This means that at any stage in the future, changes to parameter and model information in the database can be easily passed back to the placed component, synchronizing the two.

If you simply want to update parameter information, use the **Update Parameters From Database** command, available from the Schematic Editor's main **Tools** menu.

To perform a full update, including parameters, model and graphical attributes of schematic symbols, use the **Update From Libraries** command (also available from the Schematic Editor's main **Tools** menu).

In the PCB Editor, use the **Tools** » **Update From PCB Libraries** command to update placed footprints with the latest information stored in the source libraries.



For further information, refer to the *Keeping Components Up-To-Date* application note.

## Revision History

Date	Version No.	Revision
20-Jun-2006	1.0	Initial release

Software, hardware, documentation and related materials:

Copyright © 2006 Altium Limited.

All rights reserved. You are permitted to print this document provided that (1) the use of such is for personal use only and will not be copied or posted on any network computer or broadcast in any media, and (2) no modifications of the document is made. Unauthorized duplication, in whole or part, of this document by any means, mechanical or electronic, including translation into another language, except for brief excerpts in published reviews, is prohibited without the express written permission of Altium Limited. Unauthorized duplication of this work may also be prohibited by local statute. Violators may be subject to both criminal and civil penalties, including fines and/or imprisonment. Altium, Altium Designer, Board Insight, CAMtastic, CircuitStudio, Design Explorer, DXP, LiveDesign, NanoBoard, NanoTalk, Nexar, nVisage, P-CAD, Protel, SimCode, Situs, TASKING, and Topological Autorouting and their respective logos are trademarks or registered trademarks of Altium Limited or its subsidiaries. All other registered or unregistered trademarks referenced herein are the property of their respective owners and no trademark rights to the same are claimed.