Demystifying DbLink Webinar Q & A

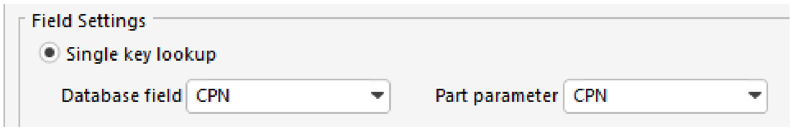
One of the most common questions asked during the webinar was, “What is the difference between the .DbLib and .DbLink files?”

Let’s start with what both files can do: both files can access an external database to update existing components on the schematic, regardless of the database’s purpose. The database’s purpose could be for purchasing, financial data, documentation, planning, etc. If a database happens to have component data that is useable for the creation of a bill of materials, these two files allow the user to gain access to the database to extract the data into the components on the schematic in the form of parameters (which is simply defined as a name and corresponding value.)

Unlike the .DbLink file, the .DbLib file is also used for the placement of components from a database onto a schematic. Note that the database must have a certain table structure for component placement capability. The graphical representations of the components (i.e., the symbols and footprints) must be listed to allow the .DbLib file to locate the appropriate library file and component within the file for placement onto the schematic.

How is a component matched up to a database component record?

Whether one uses a .DbLink or .DbLib to update components, there needs to be information that allows Altium Designer (or SOLIDWORKS PCB) to match components in the schematic to components listed in the database. This is known as the ‘key field lookup.’



Field Settings

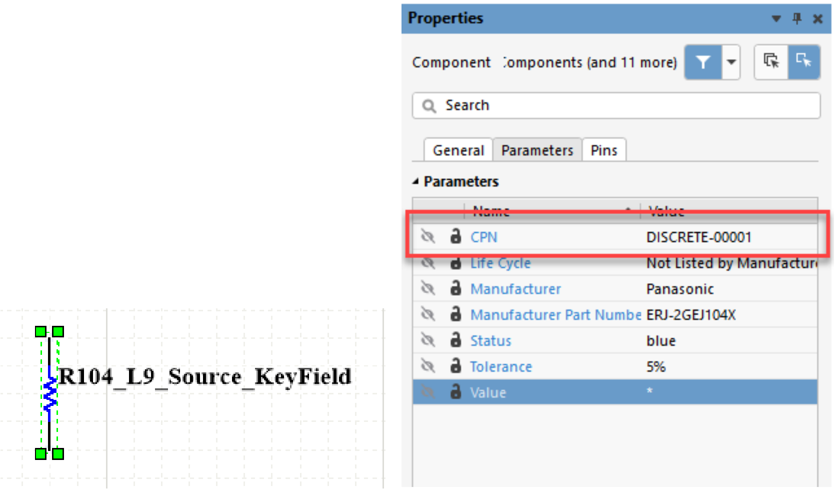
● Single key lookup

Database field: CPN

Part parameter: CPN

Figure 1: The single key look up is defined in the .DbLib and .DbLink

The key field lookup in either the .DbLink or .DbLib file needs two pieces of information. It needs to know a parameter name in the component, and a field name in the data base. The parameter name is then compared with the field name.

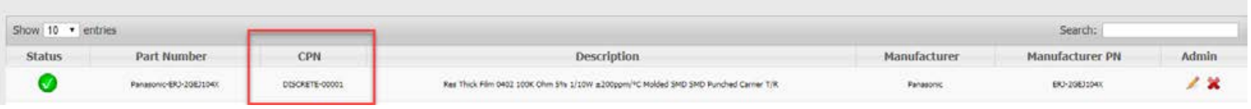


Component: :omponents (and 11 more)

Parameters

Name	Value
CPN	DISCRETE-00001
Life Cycle	Not Listed by Manufactur
Manufacturer	Panasonic
Manufacturer Part Numbe	ERJ-2GEJ104X
Status	blue
Tolerance	5%
Value	*

Figure 2: The resistor has a parameter with the name ‘CPN’ and a value of DISCETE-00001

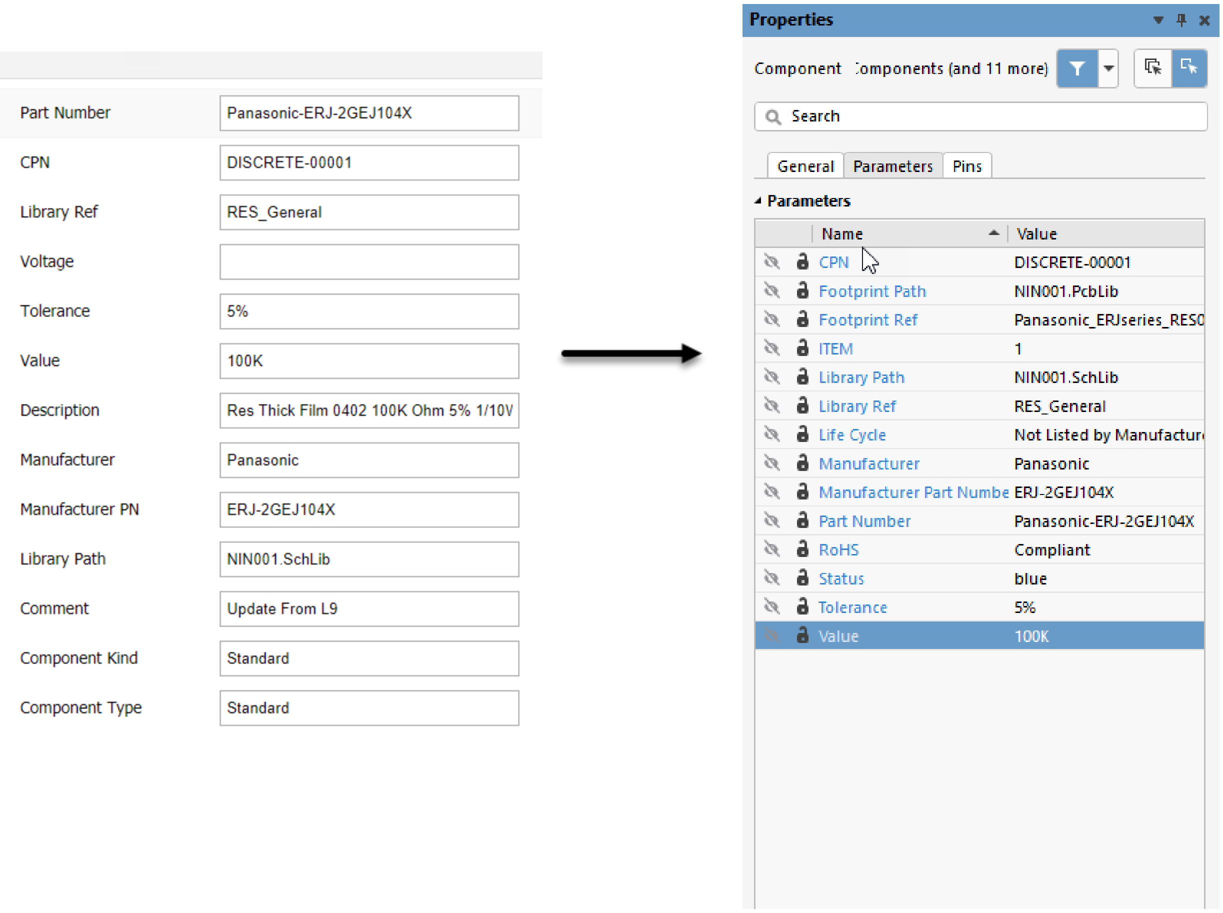


Show 10 entries Search:

Status	Part Number	CPN	Description	Manufacturer	Manufacturer PN	Admin
✓	Panasonic-ERJ-2GEJ104X	DISCRETE-00001	Res Thick Film 0402 100K Ohm 5% 1/10W ±200ppm/°C Molded SMD SMD Punched Carrier T/R	Panasonic	ERJ-2GEJ104X	✎ ✖

Figure 3: The database has a field with the name ‘CPN’, with a component that happens to have the value of DISCRETE-00001.

If the value of the parameter name matches the value of the field name, the record of information can be extracted from the database and placed into the component properties as parameters.



Part Number	Panasonic-ERJ-2GEJ104X
CPN	DISCRETE-00001
Library Ref	RES_General
Voltage	
Tolerance	5%
Value	100K
Description	Res Thick Film 0402 100K Ohm 5% 1/10V
Manufacturer	Panasonic
Manufacturer PN	ERJ-2GEJ104X
Library Path	NIN001.SchLib
Comment	Update From L9
Component Kind	Standard
Component Type	Standard

Name	Value
CPN	DISCRETE-00001
Footprint Path	NIN001.PcbLib
Footprint Ref	Panasonic_ERJseries_RESO
ITEM	1
Library Path	NIN001.SchLib
Library Ref	RES_General
Life Cycle	Not Listed by Manufactur
Manufacturer	Panasonic
Manufacturer Part Numbe	ERJ-2GEJ104X
Part Number	Panasonic-ERJ-2GEJ104X
RoHS	Compliant
Status	blue
Tolerance	5%
Value	100K

Figure 4: If the .DbLink or .DbLib file can find a match when the Tools » Update Parameters From Database is executed, data is placed in the properties of the component in the form of parameters.

For the key field lookup, the parameter in the component and the field name in the database have the same name. This does not have to be the case. In the end, for data extraction to be successful, the keyfield lookup must match the value of the ‘key’ parameter in the component to the value of the ‘key’ field in the database.

Clarification

During the webinar, the .DbLink and the .DbLib files were both pointing to the SAME database. This was done for convenience. However, in doing so, a key point was missed. In Altium Designer’s literature, when running the Tools » Update Parameters From Database”, Altium defaults to the .DbLink file followed by the .DbLib file.

When presenting the scenario in which the DbLib was “enabled” in the Libraries panel and ran the Tools » Update Parameters From Database, Altium actually used the .DbLink FIRST and then used the .DbLib SECOND.

In the second scenario, the .DbLib was disabled. Therefore, only the .DbLink was used, thus those parts that had no CPN parameter were not matched to the database.

Even if we created a third scenario in which the .DbLink was removed from the project, the .DbLib would have still found all of the components because all of the components had been placed from database library. A better demonstration would have been to create another database with the same CPN (Part Number) but with a different data set. That would have proved that the .DbLink is read prior to the .DbLib, if both exist.

Regardless, well over a half of a page of information and 20 minutes of time during the webinar was spent on trying to clarify this order of operation. To simply all of this, a feature request has been submitted allowing the user to pick which file (.DbLink or .DbLib) is to be used when using the Tools » Update Parameter From Library. Please vote for it:

https://bugcrunch.live.altium.com/#Idea/9979:

#9979: dbLink - Add a dropdown to the "Update Parameters from Database" to clarify database source

Library editor
The dbLink is a powerful, but very underutilized tool in Altium Designer. Some of the issue is the confusion that occurs when trying to use the Tools » "Update Parameters from Database" when the user has both a dbLib for a local library and a dbLink for a separate company library. This confusion requires a bit of time to sort out:

Why are these parts not being populated?
Is the function referencing the DbLib or DbLink?
Why is the wrong data being extracted into my parts?

This can all be remedied by having a dropdown in the "Update parameters from a database" dialog that allows the user to select between the dbLib and dbLink.

And since we are on the topic, please consider a checkbox that will open the log upon completion. This would be very useful when trying to determine which components were not found in the database.

Question / Comment	Answer / Response
Can .DbLink be used with the Altium Vault? For example, bringing in MFG and MPN.	No, the vault has its own abilities to manage and handle parametric data. However, the vault can treat an external database as a supplier. The general idea of the vault is to have a 'component' represented by graphical representations (a.k.a. symbols and footprint) and their main characteristics. From there, one links suppliers to these components. In this case, one could have their company data listed as a supplier.
Library attribute management is the key to success.	Indeed, it is! This goes back to what we have already said about parameters: CONSISTENCY!
Can Excel be used as a library?	Yes, it can, though this may prove a bit frustrating given that an Excel file can only be open for modification by one application at a time. If Altium is viewing, Excel will only allow you read access.
Can the .DbLink have a single key value as "Part Number"?	Yes, the .DbLink certainly can use the 'Part Number'. In fact, it's the default when the .DbLink is first opened. As mentioned in the webinar, the 'Part Number' is associated to the "Design Item ID" which is the name of the symbol. For our webinar, we used CPN (Component Part Number) just to show that another database field can be used.
If we are 100% AD users and do not have an external "vetted" library from any other department - is there an advantage to having an external DB and .DbLink or just using .DbLib (or .SVNdblib) I'm still unclear on the advantages of a .DbLink in a situation where one is already using a .DbLib. Doesn't the .DbLib already allow data updating from a database?	If the .DbLib is being used and there are no other external databases that can be tapped for useful information, there is no advantage to using .DbLink. This will only create confusion. When an update is needed from the database library, this can easily be achieved through the Tools » Update From Libraries. There is no reason to hassle with the Tools » Update Parameters from Database if a database library is being used.
Is this (dBLink) an Altium 18 thing?	No. dbLink has been around since AD6.

Since the update from database command [Tools » Update Parameters From Database] was able to pull from the active .DbLib despite the provenance being removed, what happens in the case that two different libraries have a part with the same "Part number" and the provenance has been removed?	For clarification, the term 'provenance' is a term used in database management that pertains to data's source. For our purposes, this is referring to the library from hence the component came. Remember – when we place a component, we are placing a copy of the component. By providing its provenance (or library source) it allows Altium Designer to find and update the component if Tools » Update From Libraries is used. Going back to the question, it will not matter if the parts happen to come from different libraries. The .DbLib was comparing the "Design Item ID" (a.k.a the "Part Number") with the Part Number field in the database. In all 12 cases, all the components had Design Item IDs that matched the database because they happen to have come from that same database. If I had brought a component from the miscellaneous devices library, it would have failed. This failure would not have been due to the source, rather, it would have been due to the Design Item ID not matching with any of "Part Numbers" of the database. This again goes back to the need to understand and setup the key field look up in the .DbLib file.
Where can I get directions on how to setup .DbLink to my company's database? It might be a sequel or SharePoint database but I am not sure.	To do this, the ODBC driver for the company database will need to reside on your computer. This may require some assistance from the IT folks. Once this has been setup, it a matter of creating a new .DbLink file and setting up the connection. Again, this may require the assistance of IT. NOTE: some company databases are HUGE! Make sure that you restrict your interface to specific tables of information. Otherwise, it will take an extremely long time to extract data (and lock up Altium Designer or SOLIDWORKS PCB until the extraction is complete.)
What do I need to install make our existing .DbLib work with A18?	If it worked in prior versions and now fails in AD18, it is because AD18 is a 64-bit compiled program. Therefore, it needs to access the 64-bit ODBC driver utility on your machine. There is no need to remove the 32-bit ODBC driver, especially if you are still working between AD18 and a prior version.

When you are creating a schematic, where do the parts (e.g., "resistor" symbol) come from? Then you add the CPN (in your case) to a nearly-empty part? Can the footprint come in w/ the data from the linked?	(This is in reference to the SPICE example at the end of the video) The resistors and the other components came from a symbol library (.schLib). When the component is placed, it is placed from the symbol library. Later, if an external database is available, the user will need to type in the key parameter's value so that the Tools » Update Parameters From Database can match the component with the database record, and extract the data. Tools » Update Parameters From Database will not add the footprints to the model section of the component properties; however, the parameters of the footprint will be listed in the user parameters of the component. As shown in the webinar, these parameters can be made into models using the footprint manager.
Subversion as a components database for Altium? Can you explain more or maybe make a separate webinar for that?	We referred to .DbLib files for the purposes of our conversation; however, there is also an .SVNdbLib file as well. .SVNdbLib references components that reside in an SVN (version controlled) repository rather than a file in a directory. Databases, in general, do not contain the graphics of the symbol or footprints; they only contain the path for the library file and the name of the footprint or symbol in the library. Generally speaking, .SVNdbLib adds more work to the library effort. The version control is specific to the graphic, not the component record in the database. How many times does one change a symbol or footprint? Initially, there may be corrections, however, once it is corrected, that part is not going to change thereafter. Therefore, unless there is a mandatory reason for tracking components in subversion, one should give a great deal of consideration before embarking on an SVN database.