Allegro® X Constraint Manager: Advanced Constraints Tutorial

Product Version 23.1
September 2023

Cadence Design Systems, Inc. (Cadence), 2655 Seely Ave., San Jose, CA 95134, USA.

Trademarks: Trademarks and service marks of Cadence Design Systems, Inc. contained in this document are attributed to Cadence with the appropriate symbol. For queries regarding Cadence's trademarks, contact the corporate legal department at the address shown above or call 800.862.4522.

Open SystemC, Open SystemC Initiative, OSCI, SystemC, and SystemC Initiative are trademarks or registered trademarks of Open SystemC Initiative, Inc. in the United States and other countries and are used with permission. All other trademarks are the property of their respective holders.

Restricted Permission: This publication is protected by copyright law and international treaties and contains trade secrets and proprietary information owned by Cadence. Unauthorized reproduction or distribution of this publication, or any portion of it, may result in civil and criminal penalties. Except as specified in this permission statement, this publication may not be copied, reproduced, modified, published, uploaded, posted, transmitted, or distributed in any way, without prior written permission from Cadence. Unless otherwise agreed to by Cadence in writing, this statement grants Cadence customers permission to print one (1) hard copy of this publication subject to the following conditions:

1. The publication may be used only in accordance with a written agreement between Cadence and its customer.
2. The publication may not be modified in any way.
3. Any authorized copy of the publication or portion thereof must include all original copyright, trademark, and other proprietary notices and this permission statement.
4. The information contained in this document cannot be used in the development of like products or software, whether for internal or external use, and shall not be used for the benefit of any other party, whether or not for consideration.

Disclaimer: Information in this publication is subject to change without notice and does not represent a commitment on the part of Cadence. Except as may be explicitly set forth in such agreement, Cadence does not make, and expressly disclaims, any representations or warranties as to the completeness, accuracy or usefulness of the information contained in this document. Cadence does not warrant that use of such information will not infringe any third party rights, nor does Cadence assume any liability for damages or costs of any kind that may result from use of such information. Cadence is committed to using respectful language in our code and communications. We are also active in the removal and/or replacement of inappropriate language from existing content. This product documentation may however contain material that is no longer considered appropriate but still reflects long-standing industry terminology. Such content will be addressed at a time when the related software can be updated without end-user impact.

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

Contents

Preface

Purpose of This Tutorial

The *Advanced Constraints Tutorial* provides lessons, sample design files, and text files to help you address situations in Constraint Manager where:

- There is no existing constraint for a particular requirement.
- The constraint is not a simple static value or is dependent on other values or objects.
- The validation of a constraint cannot be done using existing measurements.

The purpose of this tutorial is to introduce you to the following advanced constraints features:

- Online Formulas
- Predicates
- User-defined Properties
- User-defined Measurements.

Audience

This tutorial is intended to help users address the advanced features of Constraint Manager. To work successfully, you must have advanced knowledge of the Cadence PCB design tools.

Sample Files

Included with the tutorial are two board designs and four text files for use in performing the tasks in this tutorial. You must follow these tasks in the order in which they are presented to have a successful results.

To locate the sample files, see the *`<install_dir>/doc/cmadvcnstut/examples`* directory.

Allegro Constraint Manager: Advanced Constraint Tutorial

Introduction

The Allegro Constraint Manager Advanced Constraints Tutorial covers the following tasks:

❑Creating user-defined constraints

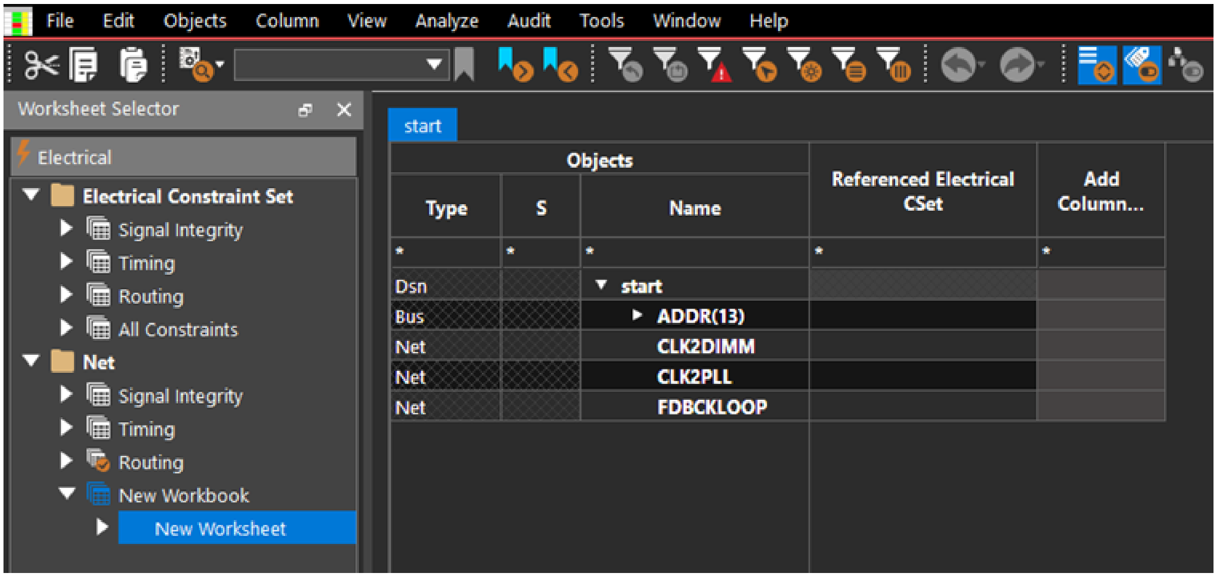
- ❑ Customizing any property or constraint with a formula
- ❑ Creating formulas with pre-defined predicates
- ❑ Using formulas with pre-defined constraints

Creating User-defined Constraints

In this first example, the requirement is to constrain nets to a particular target length with different plus and minus tolerances. Currently, this constraint does not exist in Constraint Manager as a pre-defined capability.

Using the Allegro PCB Designer license with High-Speed option:

1. Open *start.brd*.
2. Launch Constraint Manager.
3. In the *Worksheet Selector* pane, choose the *Electrical* domain, right-click on the *Net* folder, and choose *Customize Worksheet* to enter *Customize* mode.
4. Right-click on the *Net* folder again, and choose *Add New Workbook*.
5. Open the associated *New Worksheet* by double clicking on it in the *Worksheet Selector* pane.

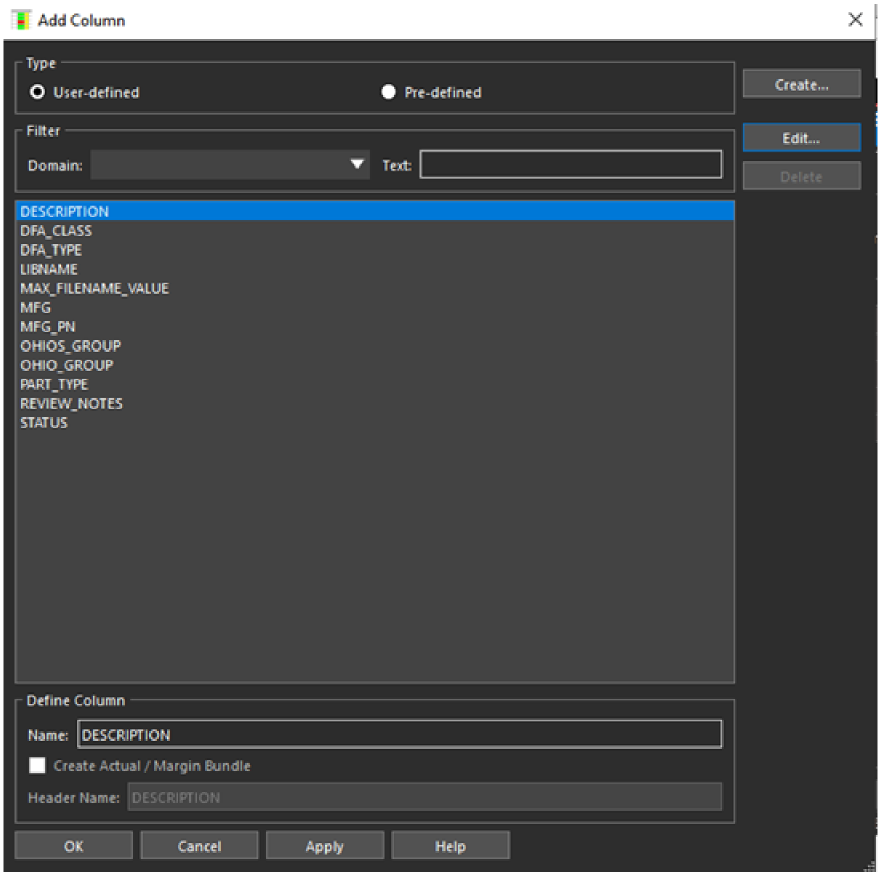


Objects			Referenced Electrical CSet	Add Column...
Type	S	Name		
*	*	*	*	*
Dsn		start		
Bus		ADDR(13)		
Net		CLK2DIMM		
Net		CLK2PLL		
Net		FDBCKLOOP		

Figure 1-1 New Worksheet

6. Click *Add Column* header in the *New Worksheet*.

7. In the *Add Column* dialog box, leave the *Type* field set to *User-defined*, and click the *Create* button.



Type

◉ User-defined ○ Pre-defined

Filter

Domain: Text:

DESCRIPTION
DFA_CLASS
DFA_TYPE
LIBNAME
MAX_FILENAME_VALUE
MFG
MFG_PN
OHIOS_GROUP
OHIO_GROUP
PART_TYPE
REVIEW_NOTES
STATUS

Define Column

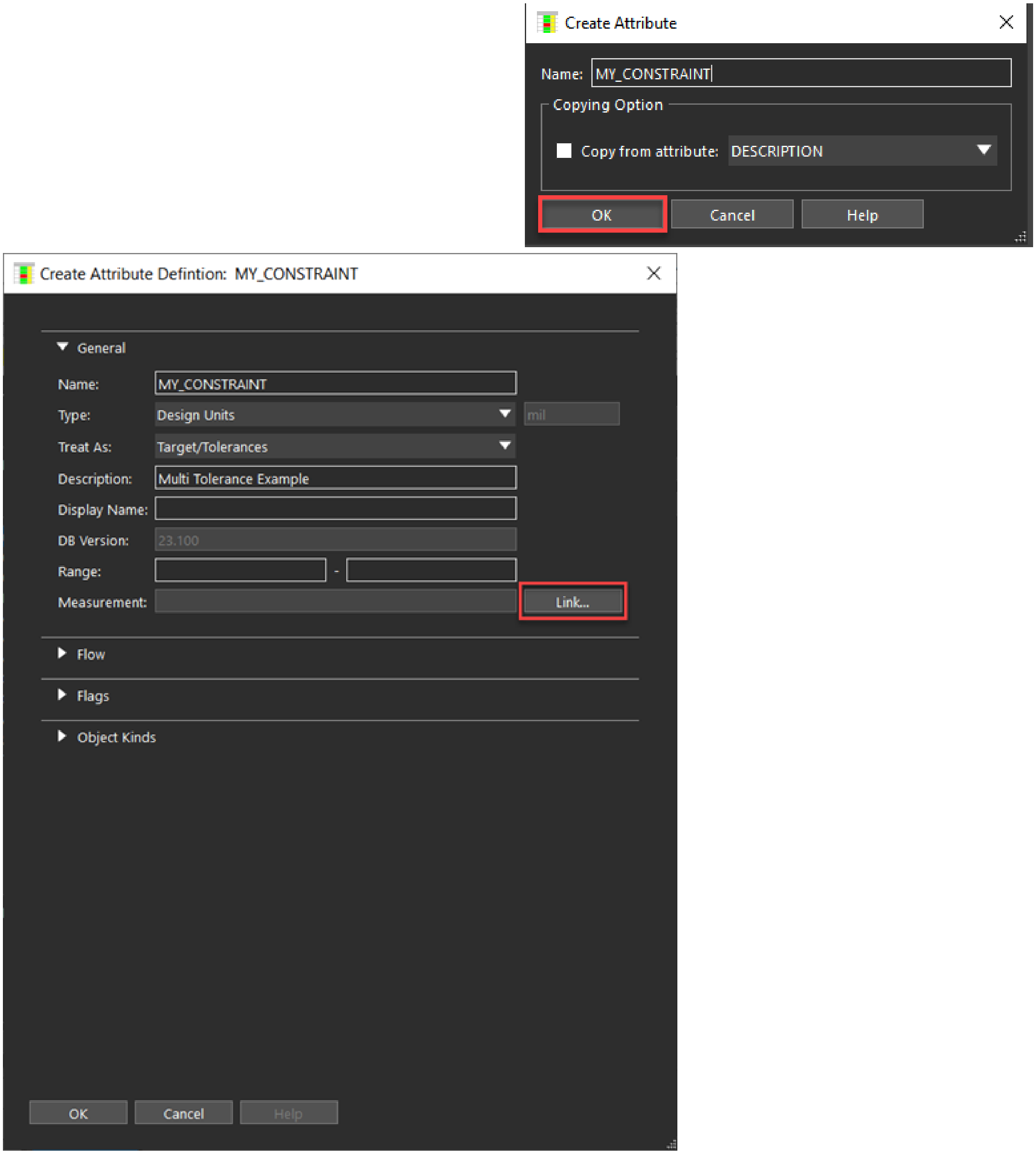
Name: DESCRIPTION

□ Create Actual / Margin Bundle

Header Name: DESCRIPTION

Figure 1-2 Add Column Dialog Box

8. Complete the *Create Attribute* and *Create Attribute Definition* dialog boxes as shown in Figure 1-3.



Create Attribute

Name: MY_CONSTRAINT

Copying Option

□ Copy from attribute: DESCRIPTION

Create Attribute Defintion: MY_CONSTRAINT

General

Name: MY_CONSTRAINT

Type: Design Units | mil

Treat As: Target/Tolerances

Description: Multi Tolerance Example

Display Name:

DB Version: 23.100

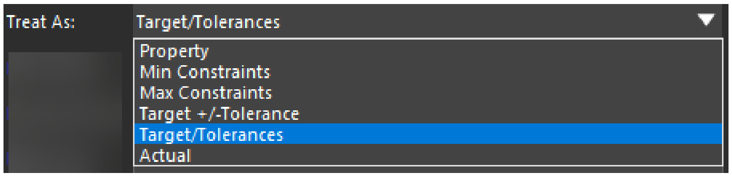
Range: -

Measurement:

Figure 1-3 Create Attribute and Create Attribute Definition Dialog Box

Make sure that there are no spaces in the *Name* field and set the *Data Type* to *Design Units*. Set the *Treat As* drop-down list as shown. Leave the *Objects* and *Flag* fields as is. The *Range* field is used in cases where the legal values for a constraint need to be defined.

Notice the six options available in the *Treat As* drop-down list. *Actual* is just a measurement with no associated constraint. *Target/Tolerances* allows for separate plus and minus tolerance values whereas *Target +/- Tolerance* allows for a single tolerance value.


Treat As: Target/Tolerances

- Property
- Min Constraints
- Max Constraints
- Target +/-Tolerance
- Target/Tolerances
- Actual

Figure 1-4 Treat As Drop-Down List

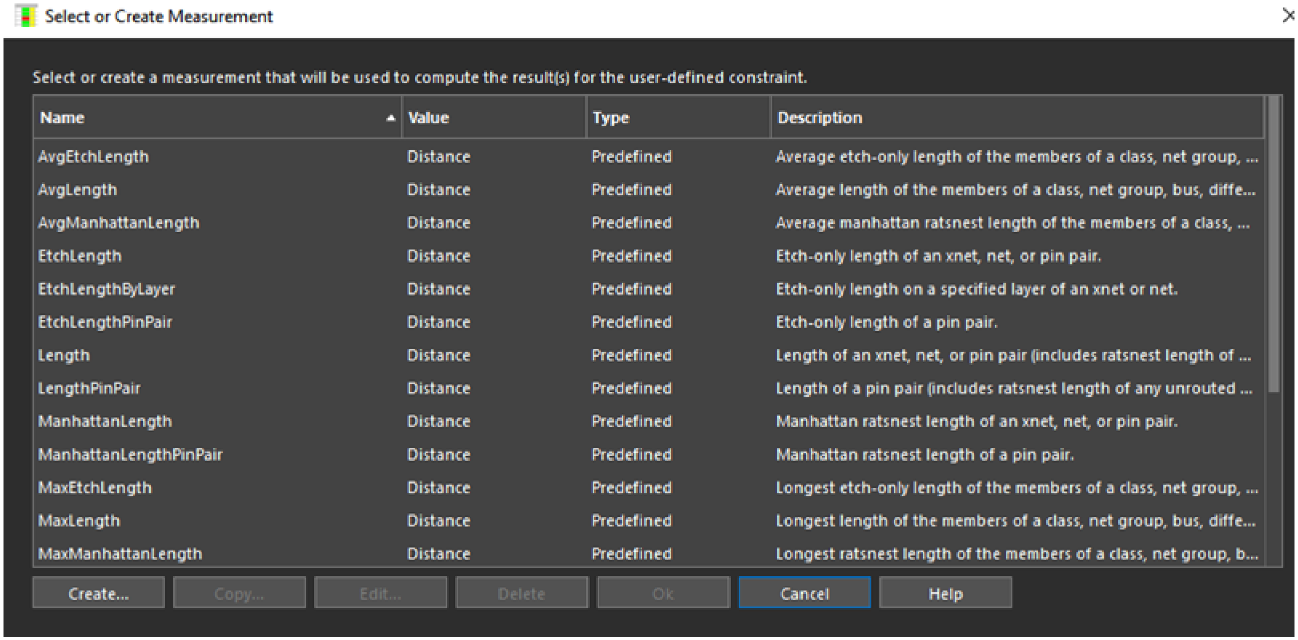
Since different plus and minus tolerance values are required for this example, choose *Target/Tolerances*.

9. Click *Link* associated with *Measurement* field.

This launches the *Select or Create Measurement* dialog box. The choices listed are based on the *Type* that you chose in the *Create Attribute Definition* dialog box (Figure 1-3). At this point, there are no user-defined measurements, so the dialog box lists the supplied pre-defined measurements.

Note: It is possible that this dialog box does not list any measurements based on the

Type you previously set.



Select or create a measurement that will be used to compute the result(s) for the user-defined constraint.

Name	Value	Type	Description
AvgEtchLength	Distance	Predefined	Average etch-only length of the members of a class, net group, ...
AvgLength	Distance	Predefined	Average length of the members of a class, net group, bus, diffe...
AvgManhattanLength	Distance	Predefined	Average manhattan ratsnest length of the members of a class, ...
EtchLength	Distance	Predefined	Etch-only length of an xnet, net, or pin pair.
EtchLengthByLayer	Distance	Predefined	Etch-only length on a specified layer of an xnet or net.
EtchLengthPinPair	Distance	Predefined	Etch-only length of a pin pair.
Length	Distance	Predefined	Length of an xnet, net, or pin pair (includes ratsnest length of ...
LengthPinPair	Distance	Predefined	Length of a pin pair (includes ratsnest length of any unrouted ...
ManhattanLength	Distance	Predefined	Manhattan ratsnest length of an xnet, net, or pin pair.
ManhattanLengthPinPair	Distance	Predefined	Manhattan ratsnest length of a pin pair.
MaxEtchLength	Distance	Predefined	Longest etch-only length of the members of a class, net group, ...
MaxLength	Distance	Predefined	Longest length of the members of a class, net group, bus, diffe...
MaxManhattanLength	Distance	Predefined	Longest ratsnest length of the members of a class, net group, b...

Figure 1-5 Select or Create Measurement Dialog Box

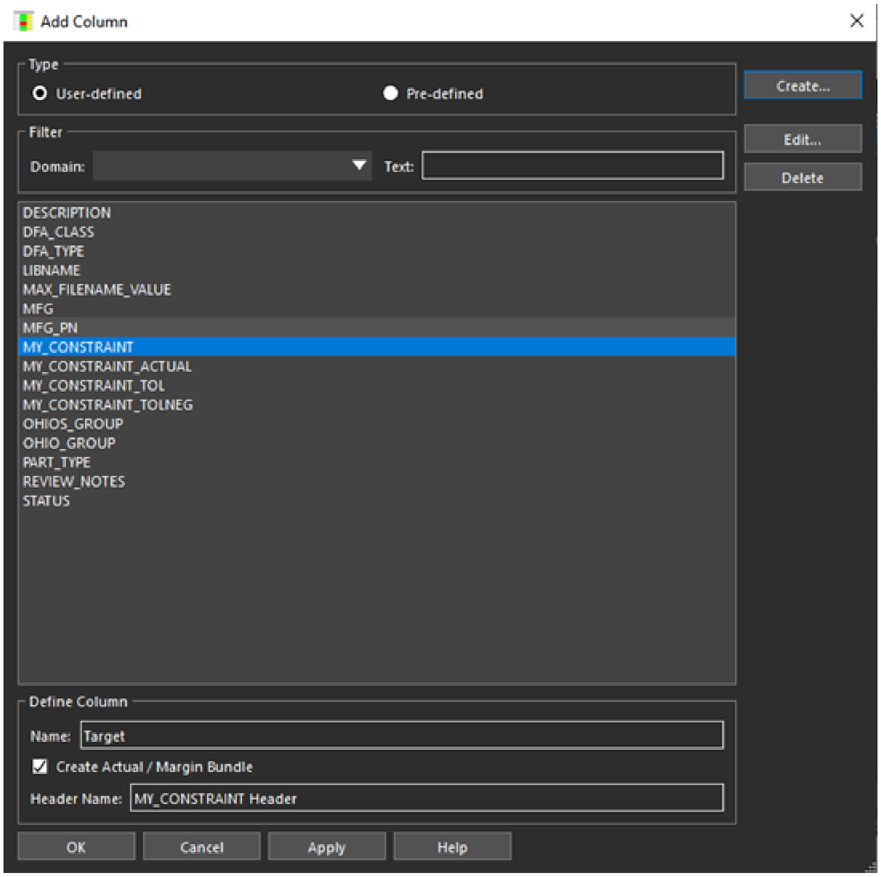
Notice that the *Copy*, *Edit*, and *Delete* buttons are grayed out, even after selecting a measurement. These buttons apply to user-defined measurements only.

10. Choose *EtchLength* from the list and click *OK*.

    The constraint value will be compared to the total etch length of the constrained net.

11. Click *OK* in the *Create Attribute Definition* dialog box.

12. In the *Add Column* dialog box (Figure 1-6) with the new columns added and the bottom of the dialog box updated, click *OK*.



DESCRIPTION
DFA_CLASS
DFA_TYPE
LIBNAME
MAX_FILENAME_VALUE
MFG
MFG_PN
MY_CONSTRAINT
MY_CONSTRAINT_ACTUAL
MY_CONSTRAINT_TOL
MY_CONSTRAINT_TOLNEG
OHIOS_GROUP
OHIO_GROUP
PART_TYPE
REVIEW_NOTES
STATUS

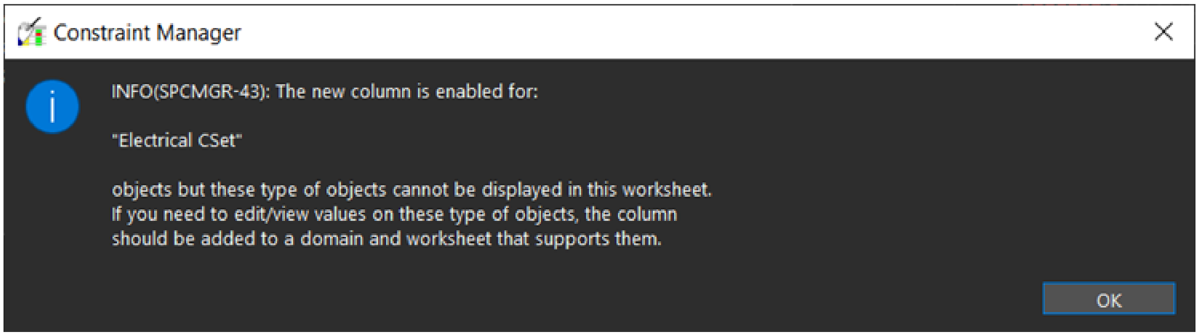
Name: Target

☑ Create Actual / Margin Bundle

Header Name: MY_CONSTRAINT Header

Figure 1-6 Add Column Dialog Box with the Newly-Added Column

13. Click *OK* if any confirmer dialog box appears.

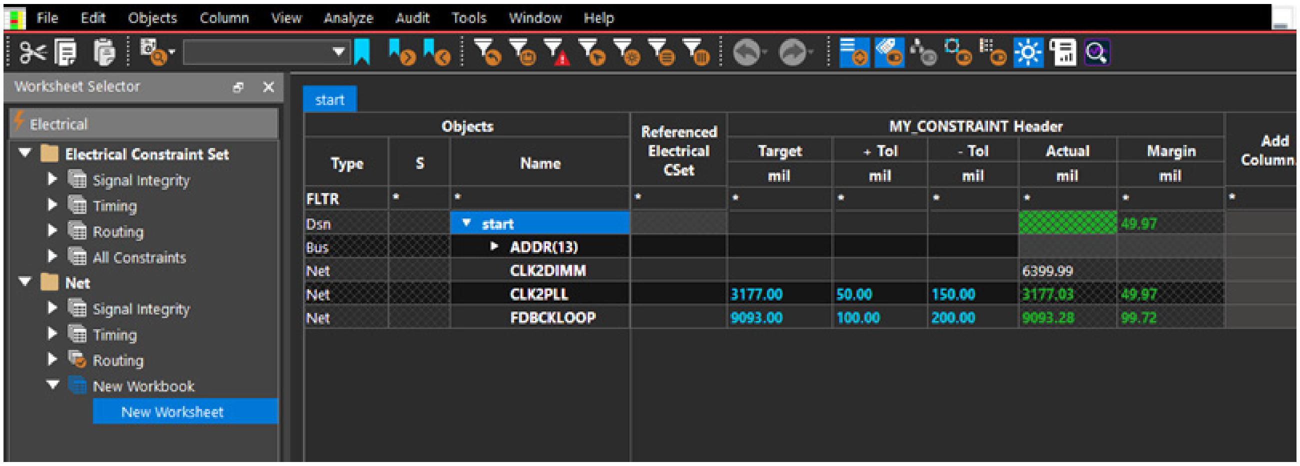


INFO(SPCMGR-43): The new column is enabled for:

"Electrical CSet"

objects but these type of objects cannot be displayed in this worksheet.
If you need to edit/view values on these type of objects, the column
should be added to a domain and worksheet that supports them.

14. In the new column, add some constraint and tolerance values to the *FDBCKLOOP* and *CLK2PLL* nets as shown Figure 1-7.



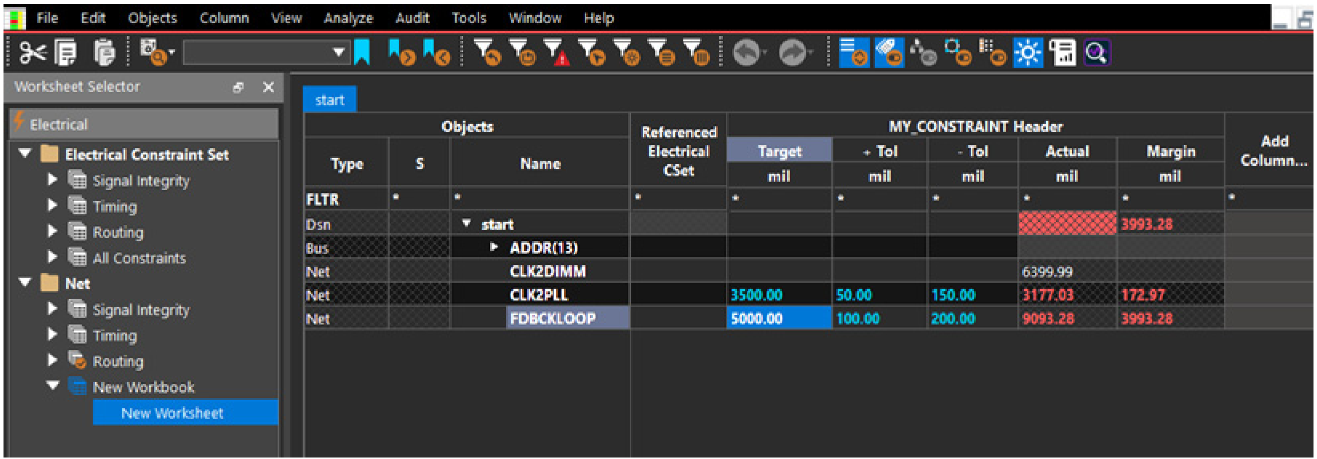
Objects			Referenced Electrical CSet	MY_CONSTRAINT Header					Add Column...
Type	S	Name		Target mil	+ Tol mil	- Tol mil	Actual mil	Margin mil	
FLTR	*	*	*	*	*	*	*	*	*
Dsn		▼ start					🟩	49.97	
Bus		▶ ADDR(13)							
Net		CLK2DIMM					6399.99		
Net		CLK2PLL		3177.00	50.00	150.00	3177.03	49.97	
Net		FDBCKLOOP		9093.00	100.00	200.00	9093.28	99.72	

Figure 1-7 Adding Constraints and Tolerance Values

15. Right-click the design name (*start)* in the *Objects* column, and choose *Analyze* or press F9.

16. Notice in Figure 1-7 that the measurement (*Actual*) is calculated even if the *Object* is not constrained. This is true for all the added user-defined *Actuals* (whether or not they are part of a user-defined constraint).

17. If the constraint values that you entered all pass, adjust one *Target* constraint value to cause an error (as shown in Figure 1-8).



Objects			Referenced Electrical CSet	MY_CONSTRAINT Header					Add Column...
Type	S	Name		Target mil	+ Tol mil	- Tol mil	Actual mil	Margin mil	
FLTR	*	*	*	*	*	*	*	*	*
Dsn		▼ start					🟥	3993.28	
Bus		▶ ADDR(13)							
Net		CLK2DIMM					6399.99		
Net		CLK2PLL		3500.00	50.00	150.00	3177.03	172.97	
Net		FDBCKLOOP		5000.00	100.00	200.00	9093.28	3993.28	

Figure 1-8 Editing Constraints and Tolerance Values

You may notice that there is no DRC in PCB Editor to indicate this situation. At this point, Pass / Fail status is indicated in Constraint Manager only.

18. Make sure to undo any etch changes that you made in PCB Editor. Verify this by re-analyzing and comparing the values in the *Actual* column of your Constraint Manager session to the values in the *Actual* column shown in Figure 1-8.

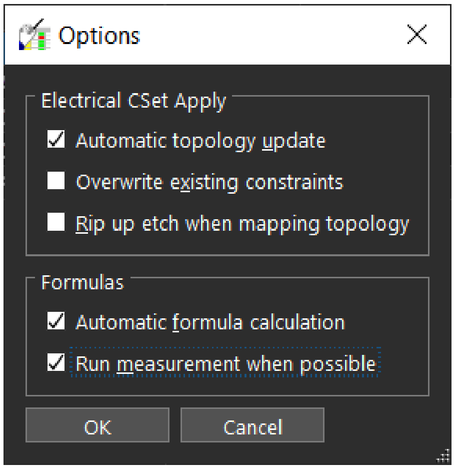
Customizing a Property or Constraint Using Formulas Created with Cell Selections

In the previous example, you created a user-defined constraint bundle where the constraint is a static value like other constraints in Constraint Manager.

But what happens if there is a case where the constraint value depends on other values? For example, what if the *FDBCKLOOP* net must equal the sum of nets *CLK2PLL* and *CLK2DIMM*? This is an example of a target constraint where the target is of the "netA + netB" type.

Continuing from the previous example:

1. Choose *Tools – Options*.
2. In the *Options* dialog box select the *Automatic formula calculation* option. Also, ensure that the *Run measurement when possible* option is selected.



Electrical CSet Apply

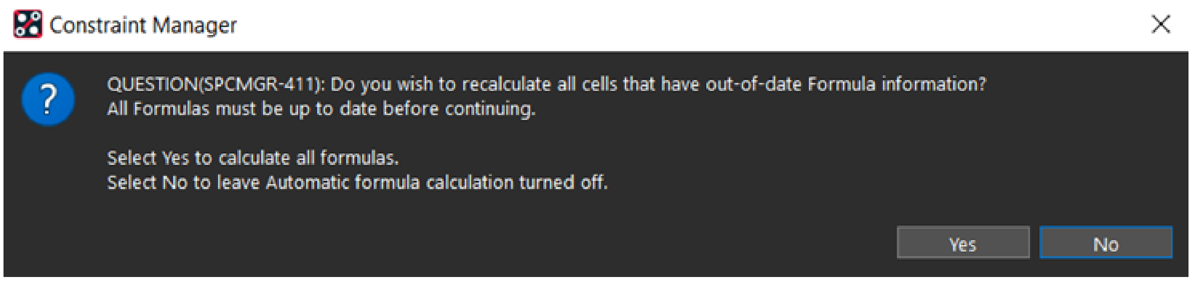
☑ Automatic topology update
☐ Overwrite existing constraints
☐ Rip up etch when mapping topology

Formulas

☑ Automatic formula calculation
☑ Run measurement when possible

Figure 1-9 Options Dialog Box

3. Click *OK* and then click *Yes* when prompted to confirm if you want to calculate all the formulas.

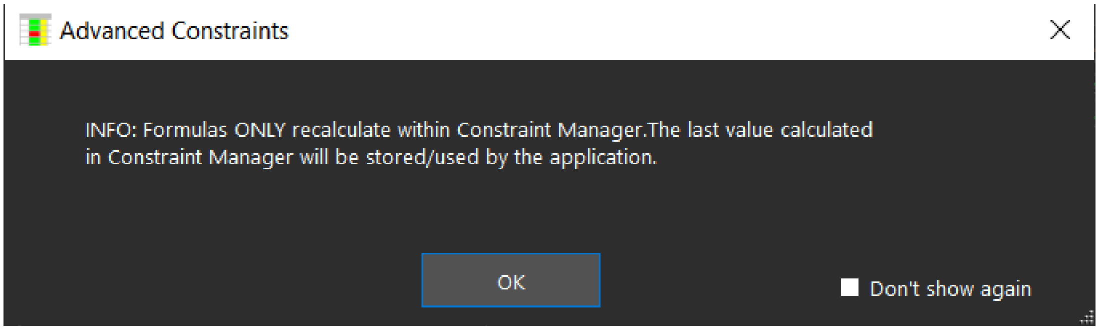


QUESTION(SPCMGR-411): Do you wish to recalculate all cells that have out-of-date Formula information?
All Formulas must be up to date before continuing.

Select Yes to calculate all formulas.
Select No to leave Automatic formula calculation turned off.

If these options are already selected the confirmer dialog box will not appear.

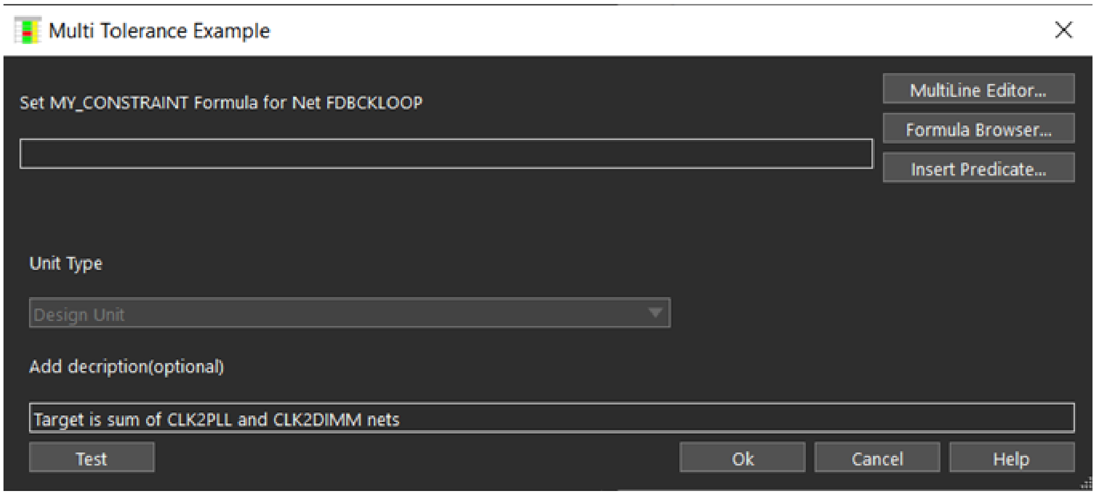
4. Right-click in the cell corresponding to the *Target* constraint for the *FDBCKLOOP* net and choose *Formula*. This launches the following confirmer dialog box.



INFO: Formulas ONLY recalculate within Constraint Manager.The last value calculated in Constraint Manager will be stored/used by the application.

5. Check the *Don't show again* box and click *OK*.

6. Type a description in the *Add description* field. Leave the dialog box open (Figure 1-10).



Set MY_CONSTRAINT Formula for Net FDBCKLOOP

Unit Type: Design Unit

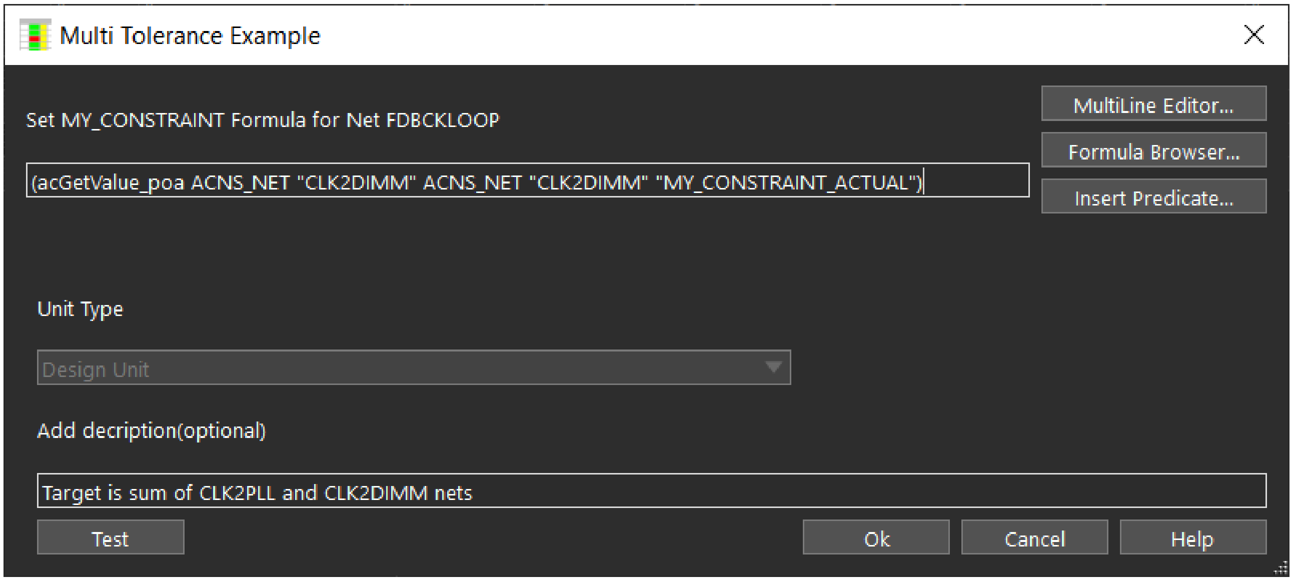
Add decription(optional): Target is sum of CLK2PLL and CLK2DIMM nets

Figure 1-10 Formula Dialog Box

Later on (see Figure 1-24), you will see that descriptions are a good way to identify particular formulas in a design for reuse.

At this point, you can generate the formula. You can type in the top fill-in field (for example, 2+3) or use the (default) Cell Selection mode.

7. Select the cell in the main Constraint Manager canvas corresponding to the *Actual* value for the *CLK2DIMM* net (should contain the value 6399.99).
   This adds the SKILL equivalent of that cell in the Formula dialog box.


Multi Tolerance Example

Set MY_CONSTRAINT Formula for Net FDBCKLOOP

```
(acGetValue_poa ACNS_NET "CLK2DIMM" ACNS_NET "CLK2DIMM" "MY_CONSTRAINT_ACTUAL")
```

Unit Type: Design Unit

Add decription(optional): Target is sum of CLK2PLL and CLK2DIMM nets

Figure 1-11 Formula Dialog Box with SKILL Code

You can also enter this text manually, although it is unlikely. Even though there is only a single cell value in the formula, a dependent constraint value has already been created. If the length of the *CLK2DIMM* net changes, the value of the constraint for the *FDBCKLOOP* net changes as well.

8. Add a "+" to the end of the text in the formula field and then select the cell corresponding to the *Actual* value for the CLK2PLL net (should contain 3177.03). The formula field now contains the string:

```
(acGetValue_poa ACNS_NET "CLK2DIMM" ACNS_NET "CLK2DIMM"
"MY_CONSTRAINT_ACTUAL")+(acGetValue_poa ACNS_NET "CLK2PLL" ACNS_NET "CLK2PLL"
"MY_CONSTRAINT_ACTUAL")
```

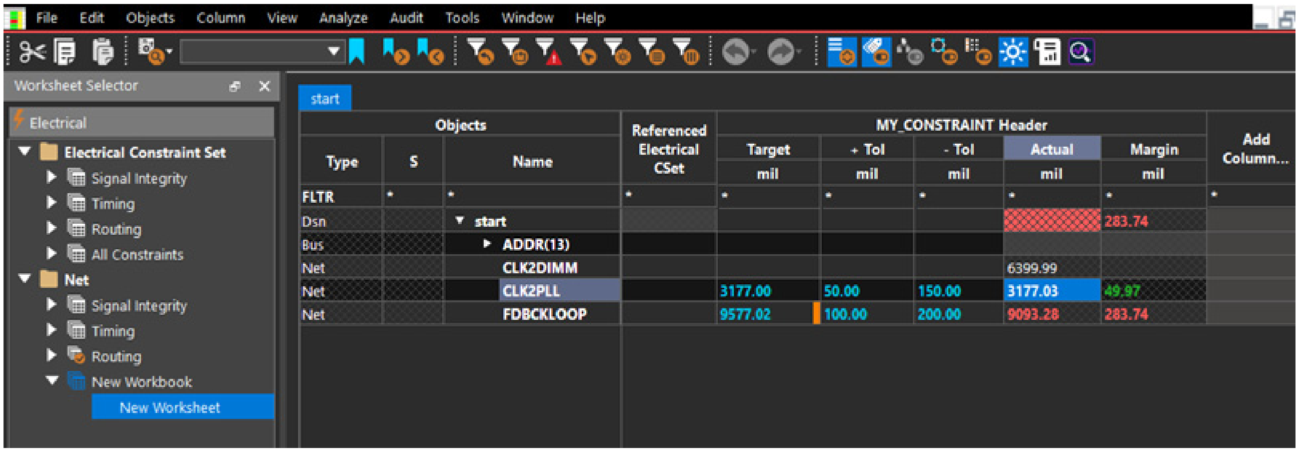
Resize the dialog box to view the text added in the formula line.

In the formula string, `acGetValue_poa` is an example of a pre-defined Predicate and the rest of the text within each set of parenthesis are parameters identifying the value for the cell that you selected.

`acGetValue` gets the value of a specific cell. There are several variants (each with a different suffix) that require different information, in this case, the **P**arent, **O**bject, and **A**ttribute.

Aside from typing a simple equation (for example, 3*4), Cell Selection is the easiest way to construct formulas.

9. Click *OK* in the *Formula* dialog box.



Objects			Referenced Electrical CSet	MY_CONSTRAINT Header					Add Column...
Type	S	Name		Target mil	+ Tol mil	- Tol mil	Actual mil	Margin mil	
FLTR	*	*	*	*	*	*	*	*	*
Dsn		▼ start						283.74	
Bus		▶ ADDR(13)							
Net		CLK2DIMM					6399.99		
Net		CLK2PLL		3177.00	50.00	150.00	3177.03	49.97	
Net		FDBCKLOOP		9577.02	100.00	200.00	9093.28	283.74	

Figure 1-12 Cell with Formula

Note: The cell now has an orange bar along the right edge indicating that the cell contains a formula. Also the value has the appropriate sum (9577.02 = 6399.99 + 3177.03).

10. In PCB Editor, use either the `slide` or `delay tune` commands to add length to either the *CLK2DIMM* net or the *CLK2PLL* net.

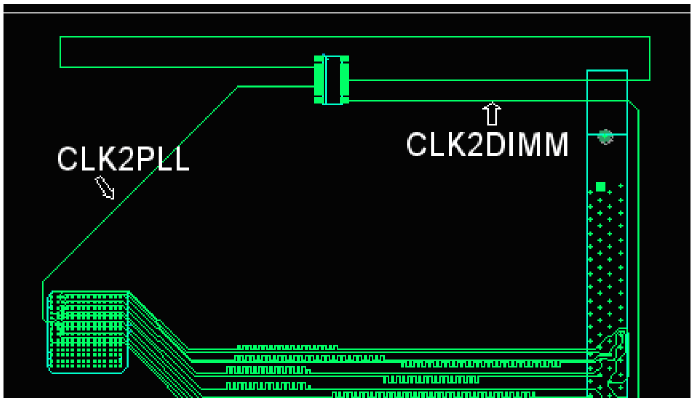
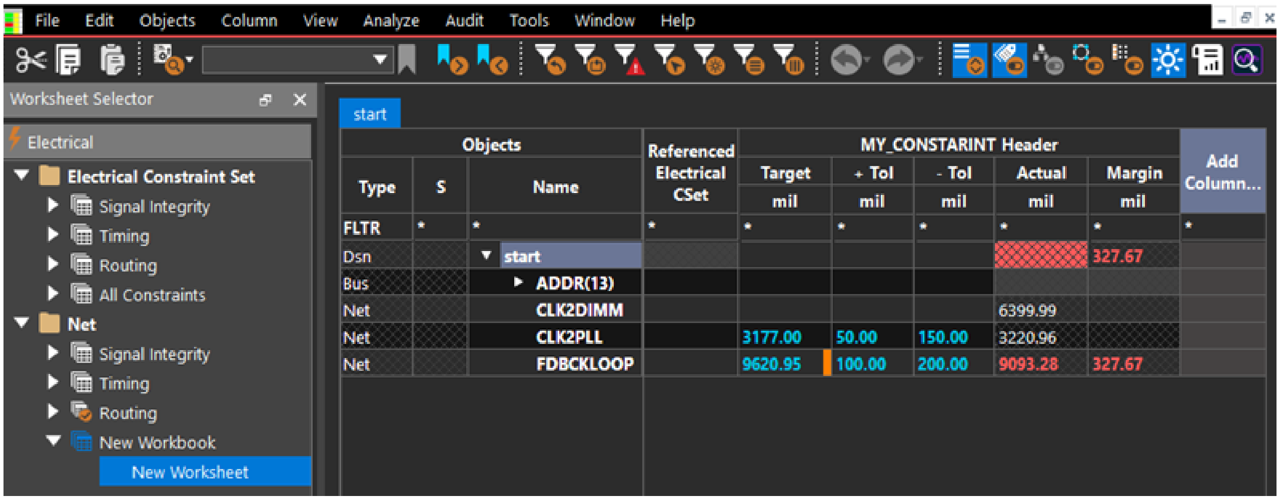


Figure 1-13 Adding Length to a Net

Notice that the *Actual* and *Margin* values in Constraint Manager corresponding to the *CLK2PLL* net that you lengthened are automatically updated. Also notice (Figure 1-14) that the cell with the formula is updated automatically with a new calculated value (*9620.95*).



Objects			Referenced Electrical CSet	MY_CONSTARINT Header					Add Column...
Type	S	Name		Target mil	+ Tol mil	- Tol mil	Actual mil	Margin mil	
FLTR	*	*	*	*	*	*	*	*	*
Dsn		▼ start						327.67	
Bus		▶ ADDR(13)							
Net		CLK2DIMM					6399.99		
Net		CLK2PLL		3177.00	50.00	150.00	3220.96		
Net		FDBCKLOOP		9620.95	100.00	200.00	9093.28	327.67	

Figure 1-14 Actual and Margin Values Updated

11. In PCB Editor, choose *Undo* to restore the lengthened net to its original state.

The Constraint Manager is closed to prevent it from displaying incorrect values caused by the *Undo*.

12. Re-open the Constraint Manager to continue.

13. Re-open the *Formula* dialog box for the same cell.

Right-click and choose from the menu or left-click in any cell that has a formula.

14. Clear the top fill-in field that has the formula.

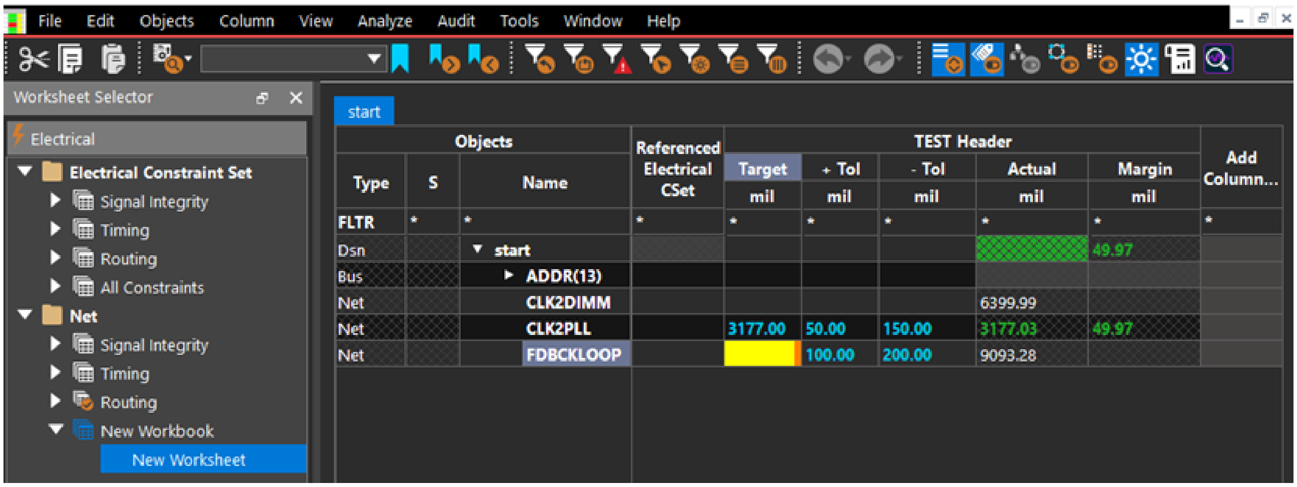
The text should be highlighted when the dialog box opens. Either press the `Delete` key or right-click in that field, and choose *Cut.*

15. Add some random characters.

The purpose is to create an error formula.

16. Click *OK* in the formula dialog box.

A yellow cell indicates an error with a formula. Notice the message in the *Status Line*.



Objects			Referenced Electrical CSet	TEST Header					Add Column...
Type	S	Name		Target	+ Tol	- Tol	Actual	Margin	
				mil	mil	mil	mil	mil	
FLTR	*	*	*	*	*	*	*	*	*
Dsn		start					🟩	49.97	
Bus		ADDR(13)							
Net		CLK2DIMM					6399.99		
Net		CLK2PLL		3177.00	50.00	150.00	3177.03	49.97	
Net		FDBCKLOOP		🟨	100.00	200.00	9093.28		

Figure 1-15 Error in the Formula

Creating Formulas with Pre-defined Predicates

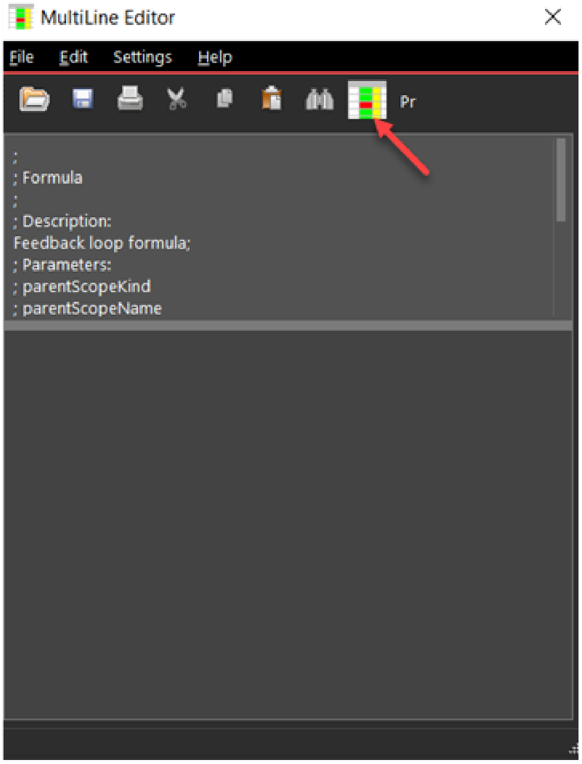
In this example, *FDBCKLOOP* represents a feedback loop for a PLL, which is used to adjust the timing for the clock signal sent to the DIMM. This feedback loop is based on the length of *CLK2PLL* (call this "A") plus the length of *CLK2DIMM* (call this "B") less the length of net *ADDR<10>* (this is "C" and is the target of the Match Group for the ADDR bus). In other words, the equation for the length of the feedback loop is:

```
(A+B)-C
```

Continuing with the results of the last example:

1. In Constraint Manager, right-click in the cell corresponding to the *Target* constraint value for *FDBCKLOOP* and choose *Clear*.
2. Right-click again in the same cell, and choose *Formula*.
3. In the *Add description* field, type *Feedback loop formula*.
4. Click the *MultiLine Editor* button.

   You can create simple formulas in the single fill-in line in the main (previous) dialog box, but the *MultiLine Editor* provides a more comprehensive editing environment.



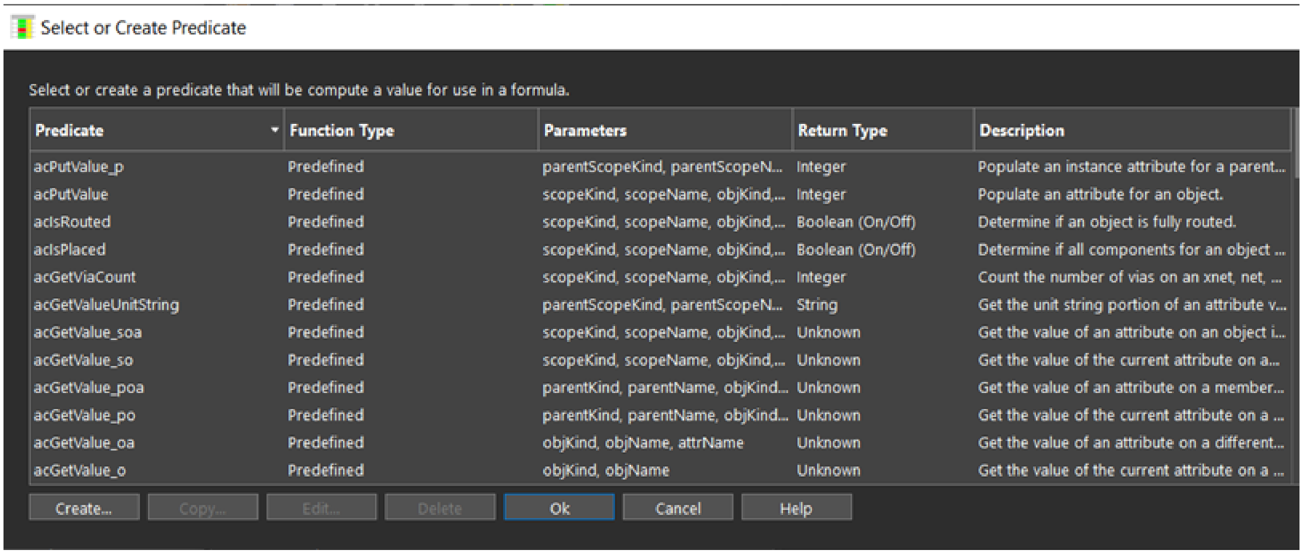
```
;
; Formula
;
; Description:
Feedback loop formula;
; Parameters:
; parentScopeKind
; parentScopeName
```

Figure 1-16 MultiLine Editor

The gray area at the top is the read-only section of the formula and lists the parameters associated with all formulas and any description that was entered in the main formula dialog box.

The icon highlighted in Figure 1-16 is the *Insert Cell Value* icon. This allows you to create formulas using cell selection, as in the previous example. You can also toggle between the single and multiline view with options under the *File* menu (the single line view is the main formula dialog box previously used).

5. Click the *Pr* (*Insert Predicate*) icon at the end of the toolbar. This icon opens the *Select or Create Predicate* dialog box.

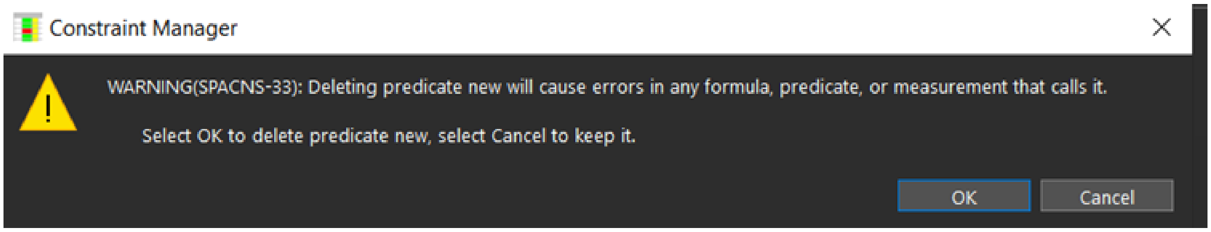


Select or create a predicate that will be compute a value for use in a formula.

Predicate	Function Type	Parameters	Return Type	Description
acPutValue_p	Predefined	parentScopeKind, parentScopeN...	Integer	Populate an instance attribute for a parent...
acPutValue	Predefined	scopeKind, scopeName, objKind,...	Integer	Populate an attribute for an object.
acIsRouted	Predefined	scopeKind, scopeName, objKind,...	Boolean (On/Off)	Determine if an object is fully routed.
acIsPlaced	Predefined	scopeKind, scopeName, objKind,...	Boolean (On/Off)	Determine if all components for an object ...
acGetViaCount	Predefined	scopeKind, scopeName, objKind,...	Integer	Count the number of vias on an xnet, net, ...
acGetValueUnitString	Predefined	parentScopeKind, parentScopeN...	String	Get the unit string portion of an attribute v...
acGetValue_soa	Predefined	scopeKind, scopeName, objKind,...	Unknown	Get the value of an attribute on an object i...
acGetValue_so	Predefined	scopeKind, scopeName, objKind,...	Unknown	Get the value of the current attribute on a...
acGetValue_poa	Predefined	parentKind, parentName, objKind...	Unknown	Get the value of an attribute on a member...
acGetValue_po	Predefined	parentKind, parentName, objKind...	Unknown	Get the value of the current attribute on a ...
acGetValue_oa	Predefined	objKind, objName, attrName	Unknown	Get the value of an attribute on a different...
acGetValue_o	Predefined	objKind, objName	Unknown	Get the value of the current attribute on a ...

Figure 1-17 Select or Create Predicate Dialog Box

Your list of Predicates may be different from the list shown here. You can copy, edit, or delete a user-defined predicate from this list. When you select a user-defined predicate from the list and click the *Delete* button, the following warning message pops-up. If you select *OK*, the predicate is deleted from the list.



WARNING(SPACNS-33): Deleting predicate new will cause errors in any formula, predicate, or measurement that calls it.

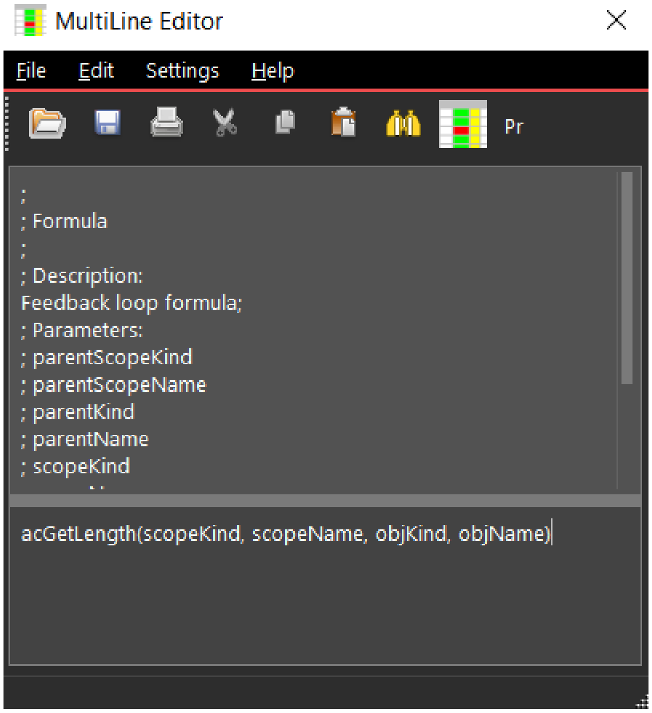
Select OK to delete predicate new, select Cancel to keep it.

Predicates are the building blocks of formulas and user-defined Measurements. The complete list of Predicates is presented.

Predicates are similar to Measurements in that they return a particular value from the design. They differ in several ways:

- Predicates only return a single value; Measurements can return many values.
- Measurements are used for *Actuals*; Predicates are used for formulas, measurements, or other predicates.
- The list of Predicates presented is complete; the list for measurements is based on the *Data Type* of the *Actual.*

6. Choose the `acGetLength` predicate in the list and then click *OK*. See Figure 1-17.



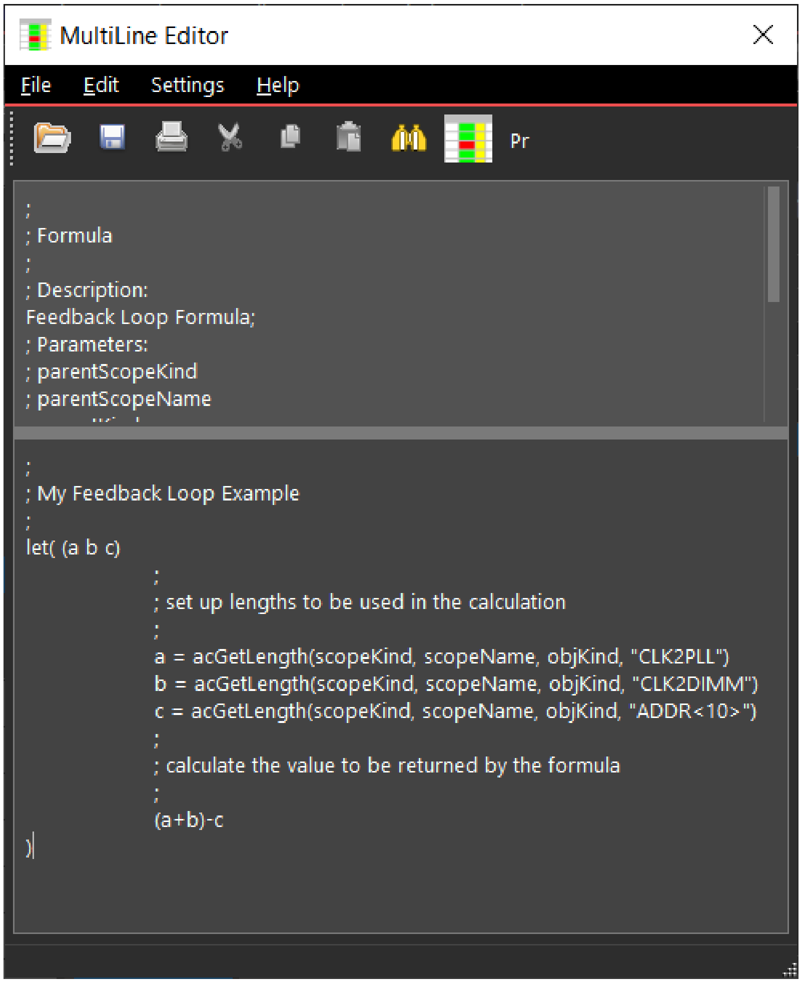
```
;
; Formula
;
; Description:
Feedback loop formula;
; Parameters:
; parentScopeKind
; parentScopeName
; parentKind
; parentName
; scopeKind
```

```
acGetLength(scopeKind, scopeName, objKind, objName)
```

Figure 1-18 MultiLine Editor

The constraint is based on length (the length of A + the length …) so a predicate for `GetLength` makes sense.

7. Use *Copy* and *Paste* or re-insert the same predicate twice and then edit the resulting *objName* parameters for each to create the formula shown below.



```
;
; Formula
;
; Description:
Feedback Loop Formula;
; Parameters:
; parentScopeKind
; parentScopeName
```

```
;
; My Feedback Loop Example
;
let( (a b c)
            ;
            ; set up lengths to be used in the calculation
            ;
            a = acGetLength(scopeKind, scopeName, objKind, "CLK2PLL")
            b = acGetLength(scopeKind, scopeName, objKind, "CLK2DIMM")
            c = acGetLength(scopeKind, scopeName, objKind, "ADDR<10>")
            ;
            ; calculate the value to be returned by the formula
            ;
            (a+b)-c
)
```

Figure 1-19 Feedback Loop Example

If you prefer, load the *fb_loop_example.txt.* Choose *File – Load from file* menu command and browse to the file in *<install_dir>/doc/cmadvcnstut/examples*.

For this formula, the `acGetLength` predicate was used. Notice that it has four parameters and for this case, only the *objName* parameter was changed to the net name of interest. Depending on the specific application, you may need to edit additional parameters, most likely *scope* parameters for system configurations and *parent* parameters for *Match Groups*.

Also notice that variables were used to clean up the display. This formula works just as well when written as:

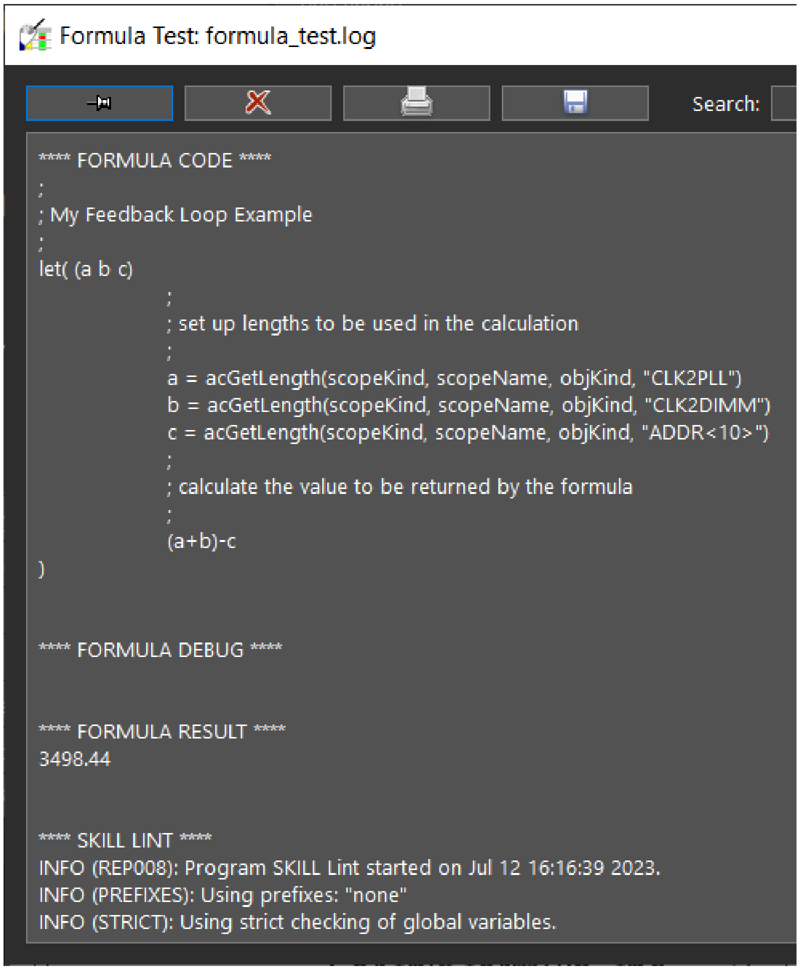
```
(acGetLength(scopeKind, scopeName, objKind, "CLK2PLL") +
acGetLength(scopeKind, scopeName, objKind, "CLK2DIMM")) -
acGetLength(scopeKind, scopeName, objKind, "ADDR<10>")
```

Tip

The entire function is wrapped in a `let` statement. Use the `let` statement any time that variables are used. These variables are being protected from becoming global and the potential side effects from having them accidentally accessed elsewhere.

8. From the menu, choose *File – Test*.

Choosing this menu item is the same as clicking the *Test* button in the lower-left corner of the main (single line) *Formula* dialog box (Figure 1-10).



```
**** FORMULA CODE ****
;
; My Feedback Loop Example
;
let( (a b c)
        ;
        ; set up lengths to be used in the calculation
        ;
        a = acGetLength(scopeKind, scopeName, objKind, "CLK2PLL")
        b = acGetLength(scopeKind, scopeName, objKind, "CLK2DIMM")
        c = acGetLength(scopeKind, scopeName, objKind, "ADDR<10>")
        ;
        ; calculate the value to be returned by the formula
        ;
        (a+b)-c
)


**** FORMULA DEBUG ****


**** FORMULA RESULT ****
3498.44


**** SKILL LINT ****
INFO (REP008): Program SKILL Lint started on Jul 12 16:16:39 2023.
INFO (PREFIXES): Using prefixes: "none"
INFO (STRICT): Using strict checking of global variables.
```

Figure 1-20 Formula Test Dialog Box

In the *Formula Test* dialog box, there are four main sections of the output:

- A listing of the Formula itself.
- A debug message section, which is currently blank.
- The formula result. In this case, verify that 3498.44 is the expected result.
- SKILL LINT output to further check the SKILL code for possible errors.

Tip

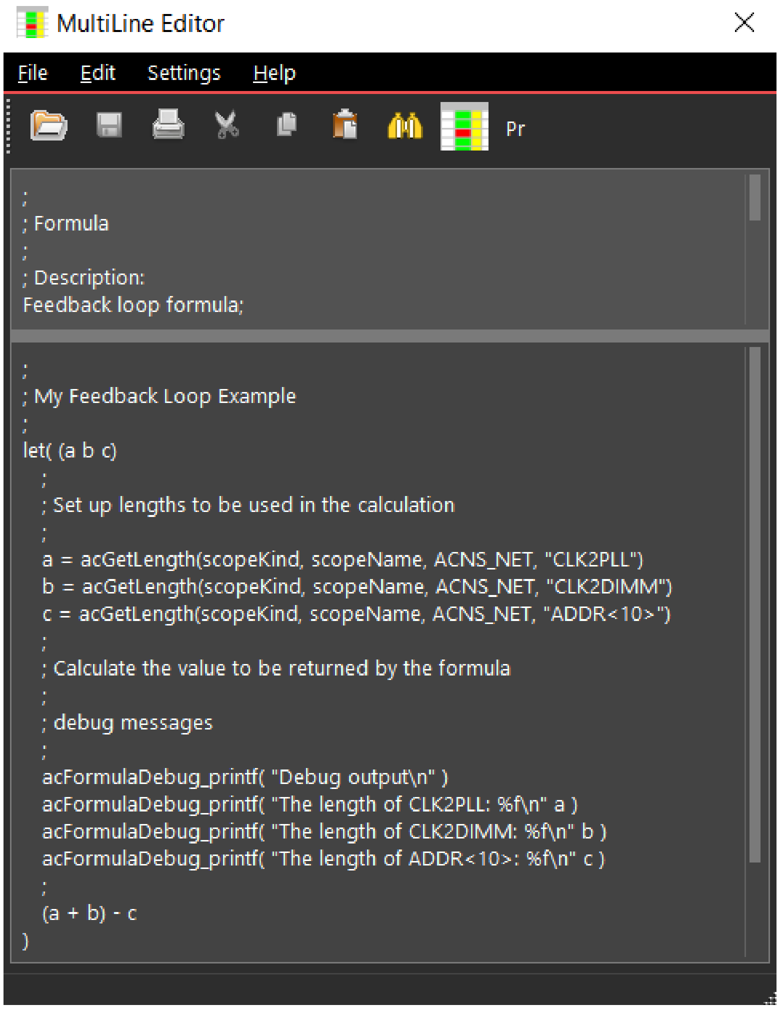
When you create any custom code for a formula, always test the code, no matter how simple the code seems.

9. Close the *Formula Test* dialog box.

As mentioned above, verify that the result is as expected by checking the lengths and doing the math. However, there is another way to do this earlier in the process.

10. In the MultiLine Editor, use *File – Load from file*, and load the *debug_example.txt* file.

Notice the added `acFormulaDebug_printf` predicates.

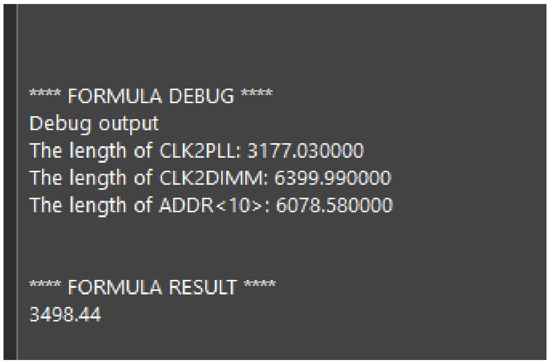


```
;
; Formula
;
; Description:
Feedback loop formula;
```

```
;
; My Feedback Loop Example
;
let( (a b c)
   ;
   ; Set up lengths to be used in the calculation
   ;
   a = acGetLength(scopeKind, scopeName, ACNS_NET, "CLK2PLL")
   b = acGetLength(scopeKind, scopeName, ACNS_NET, "CLK2DIMM")
   c = acGetLength(scopeKind, scopeName, ACNS_NET, "ADDR<10>")
   ;
   ; Calculate the value to be returned by the formula
   ;
   ; debug messages
   ;
   acFormulaDebug_printf( "Debug output\n" )
   acFormulaDebug_printf( "The length of CLK2PLL: %f\n" a )
   acFormulaDebug_printf( "The length of CLK2DIMM: %f\n" b )
   acFormulaDebug_printf( "The length of ADDR<10>: %f\n" c )
   ;
   (a + b) - c
)
```

Figure 1-21 Loading the debug_example File

11. Choose *File – Test*.



```
**** FORMULA DEBUG ****
Debug output
The length of CLK2PLL: 3177.030000
The length of CLK2DIMM: 6399.990000
The length of ADDR<10>: 6078.580000


**** FORMULA RESULT ****
3498.44
```

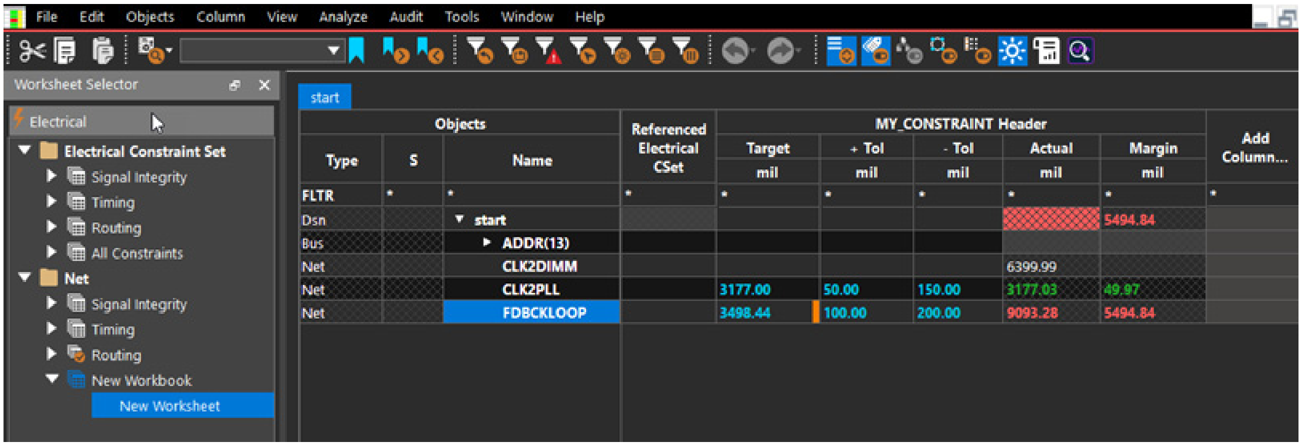
Figure 1-22 FORMULA DEBUG Section in the Formula Test Dialog Box

The added print commands output text in the FORMULA DEBUG section also aid in verifying the RESULT.

12. Close the *Formula Test* dialog box.

13. Close the MultiLine Editor window and click *Save* if prompted.

The value of the formula in Constraint Manager should be 3498.44.



Objects			Referenced Electrical CSet	MY_CONSTRAINT Header					Add Column...
Type	S	Name		Target mil	+ Tol mil	- Tol mil	Actual mil	Margin mil	
FLTR	*	*	*	*	*	*	*	*	*
Dsn		▾ start					▨	5494.84	
Bus		▸ ADDR(13)							
Net		CLK2DIMM					6399.99		
Net		CLK2PLL		3177.00	50.00	150.00	3177.03	49.97	
Net		FDBCKLOOP		3498.44	100.00	200.00	9093.28	5494.84	

Figure 1-23 Updated Value in the Worksheet

Using Formulas with Pre-defined Constraints

Until now, the focus has been on user-defined constraints. While this opens a new set of possibilities for constraining objects and validating the results in Constraint Manager, many PCB designers prefer to work in the PCB Editor canvas with real-time feedback.

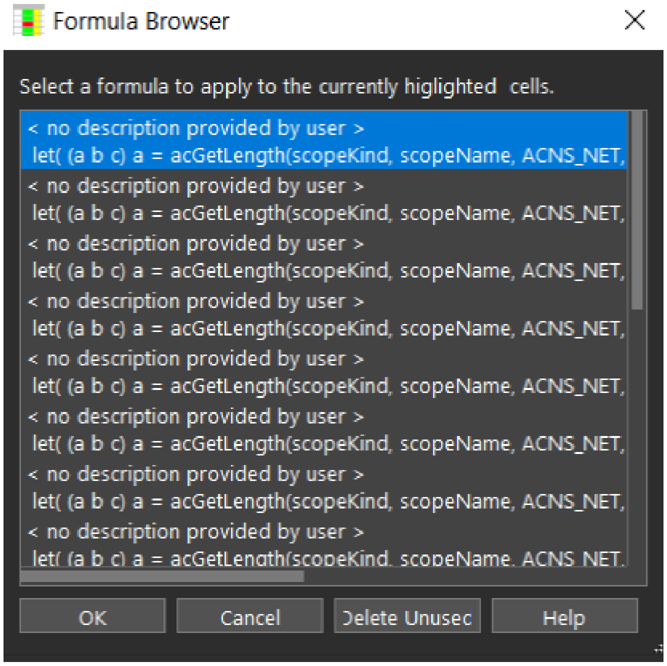
1. Open the *Net – Routing – Min/Max Propagation Delays* worksheet in the *Electrical Domain*.

   You will leverage the formula that you just created. While the formula returns a single value based on the requirements (the length of (a+b) - c), it does not make sense as both the minimum and maximum constraint value. In other words, there needs to be some tolerance which was handled in the user-defined constraint previously used. For this example, you will add the tolerance directly to the formula.

2. Right-click on the *Min Prop Delay* cell for *FDBCKLOOP*, and choose *Formula*.

3. In the *Formula* dialog box, choose *Design Unit* from the *Select Unit Type* drop-down list.

4. Click the *Formula Browser* button.



Select a formula to apply to the currently higlighted cells.

```
< no description provided by user >
let( (a b c) a = acGetLength(scopeKind, scopeName, ACNS_NET,
< no description provided by user >
let( (a b c) a = acGetLength(scopeKind, scopeName, ACNS_NET,
< no description provided by user >
let( (a b c) a = acGetLength(scopeKind, scopeName, ACNS_NET,
< no description provided by user >
let( (a b c) a = acGetLength(scopeKind, scopeName, ACNS_NET,
< no description provided by user >
let( (a b c) a = acGetLength(scopeKind, scopeName, ACNS_NET,
< no description provided by user >
let( (a b c) a = acGetLength(scopeKind, scopeName, ACNS_NET,
< no description provided by user >
let( (a b c) a = acGetLength(scopeKind, scopeName, ACNS_NET,
< no description provided by user >
let( (a b c) a = acGetLength(scopeKind, scopeName, ACNS_NET,
```

Figure 1-24 Formula Browser Dialog Box

Your list may be different. You can delete an unused formula from this list. When you click the *Delete Unused* button, a warning message will pop-up. If you click *OK* to proceed with the delete operation, any formula in the current design that is not assigned to a cell will be deleted.

Note: Note that formulas loaded from the `.wcf` file are considered global and are not deleted by this command.

The *Formula Browser* dialog box is one way to use existing formulas in the design. Notice that as you hover over each formula, a tool tip window appears showing the full formula.

5. Choose one of the Feedback Loop examples and click *OK* in the *Formula Browser* dialog box (the formula from the previous exercise).
6. Click the *MultiLine Editor* button in the main Formula dialog box, and change the last line of the formula to:

   ```
   ((a + b) - c) - 25
   ```

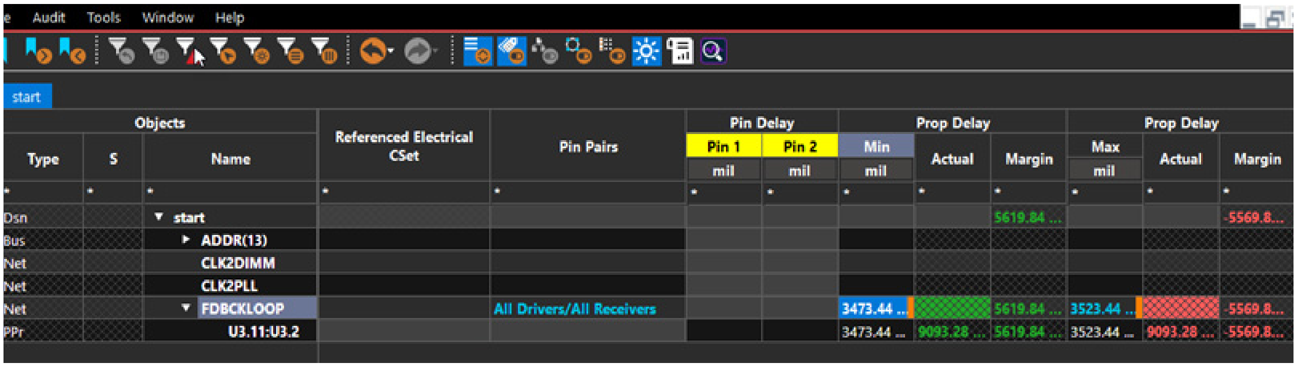
   The *Min Prop Delay* constraint value is set to 25 mils (design units) less than the target length.
7. Save and close the MultiLine Editor.
8. Right-click on the *Min Prop Delay* cell for *FDBCKLOOP*, and choose *Copy* (this is the cell in which you just entered the formula).
9. Right-click in the *Max Prop Delay* cell for *FDBCKLOOP* and choose *Paste Special*.

   Ensure that *Select paste mode* is set to *Formula*, which is the default action.
10. Make sure that you select the *Formula* radio button and click *OK* in the Paste Special dialog box.
11. Edit the formula for the *Max Prop Delay* and adjust the last line of the formula to:

    ```
    ((a + b) - c) + 25
    ```

    Notice that the *Formula* dialog box opens in the MultiLine Editor. It opens in whichever dialog box you last used in creating the formula.
12. Save and close the MultiLine Editor.

Your view should look similar to what is shown in Figure 1-25.



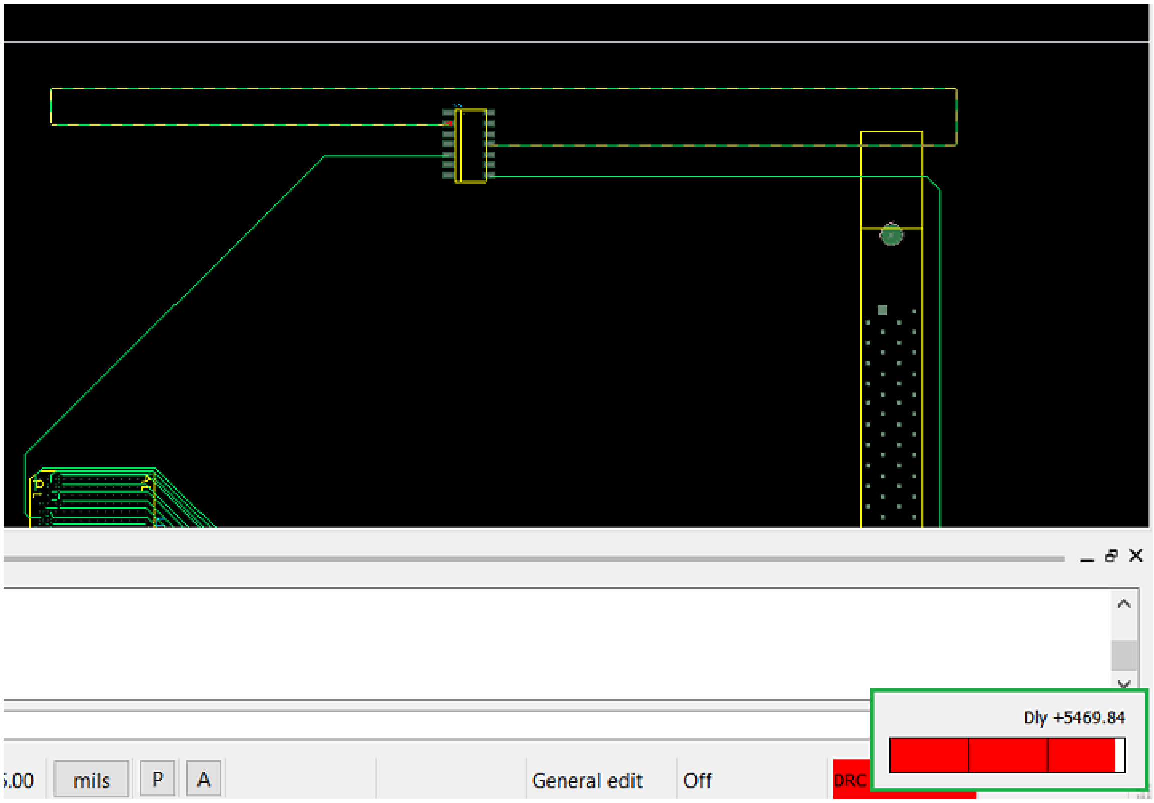
Objects			Referenced Electrical CSet	Pin Pairs	Pin Delay		Prop Delay			Prop Delay		
Type	S	Name			Pin 1	Pin 2	Min	Actual	Margin	Max	Actual	Margin
					mil	mil	mil			mil		
*	*	*	*	*	*	*	*	*	*	*	*	*
Dsn		▼ start							5619.84 ...			-5569.8...
Bus		▸ ADDR(13)										
Net		CLK2DIMM										
Net		CLK2PLL										
Net		▼ FDBCKLOOP		All Drivers/All Receivers			3473.44 ...		5619.84 ...	3523.44 ...		-5569.8...
PPr		U3.11:U3.2					3473.44 ...	9093.28 ...	5619.84 ...	3523.44 ...	9093.28 ...	-5569.8...

Figure 1-25 View in Constraint Manager

Look for the values 3473.44 and 3523.44 in the *Min* and *Max* formula cells, respectively.

13. In PCB Editor, run the `slide` command for the *FDBCKLOOP* net (the top net in the design) and notice that the Heads-up Display can be used.

Figure 1-26 Sliding the FDBCKLOOP Net



14. Cancel the `slide` command (do not commit to any etch changes).

Creating User-defined Measurements to Populate User-defined Actuals

User-defined Measurements and Predicates represent the final pieces of the customization process.

- A user-defined Predicate is a SKILL function with arbitrary parameters that returns a single value for use in a Formula or Measurement.
- A user-defined Measurement is a SKILL function with a pre-determined set of input parameters (the object being measured) that must update result information and return success or failure. Result information includes pushing of the *Actual* values. You can push the *Actual* values on the object being measured, or you can push the values on result objects that you associate with the object being measured.

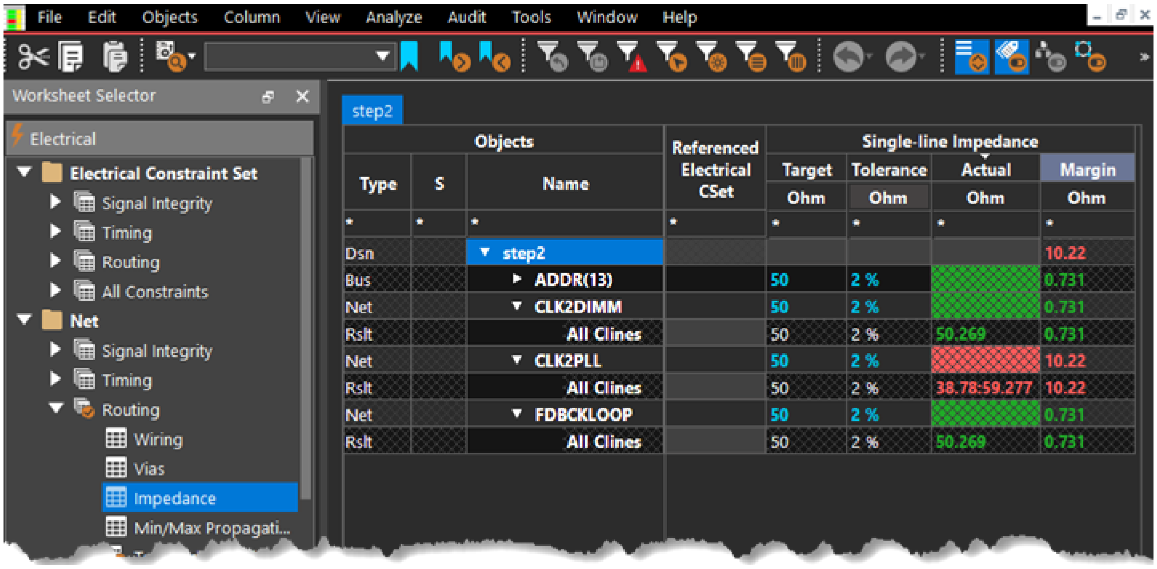
With the ability to code your own Measurements to associate with *Actuals* either as is or in user-defined Constraints coupled with the ability to make user-defined Predicates as the building blocks for both Measurements and Formulas, you can theoretically create any scenario in Constraint Manager.

For this next example, examine adding an *Actual* (associated with a user-defined Measurement) to aid in debugging an existing constraint.

1. In PCB Editor, open *step2.brd*.

   Do not save the current design.

2. Launch Constraint Manager and open the *Net – Routing – Impedance* worksheet. Right-click on the design name*,* and choose *Analyze* to generate results.



Objects			Referenced Electrical CSet	Single-line Impedance			
Type	S	Name		Target	Tolerance	Actual	Margin
				Ohm	Ohm	Ohm	Ohm
*	*	*	*	*	*	*	*
Dsn		▼ step2					10.22
Bus		▶ ADDR(13)		50	2 %	🟩	0.731
Net		▼ CLK2DIMM		50	2 %	🟩	0.731
Rslt		All Clines		50	2 %	50.269	0.731
Net		▼ CLK2PLL		50	2 %	🟥	10.22
Rslt		All Clines		50	2 %	38.78:59.277	10.22
Net		▼ FDBCKLOOP		50	2 %	🟩	0.731
Rslt		All Clines		50	2 %	50.269	0.731

Figure 1-27 Impedance Worksheet

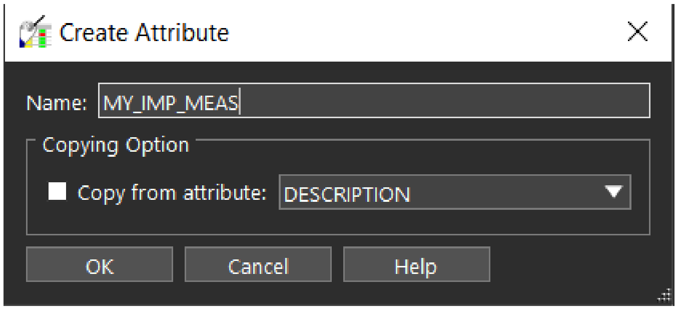
The *CLK2PLL* net has at least two errors since the *Actual* value is reporting a range of results from 38.78 to 59.277 Ohms and both values exceed the target +/- the 2% tolerance. Adding a user-defined measurement to this worksheet can aid in debugging the errors for this net.

3. Right-click in the *Worksheet Selector,* and choose *Customize Worksheet*.

4. Right-click on the *Net – Impedance* worksheet and choose *Add Column*.

5. Click the *Create* button in the *Add Column* dialog box.

6. Fill in the *Create Attribute* dialog box as shown in Figure 1-28.


Create Attribute

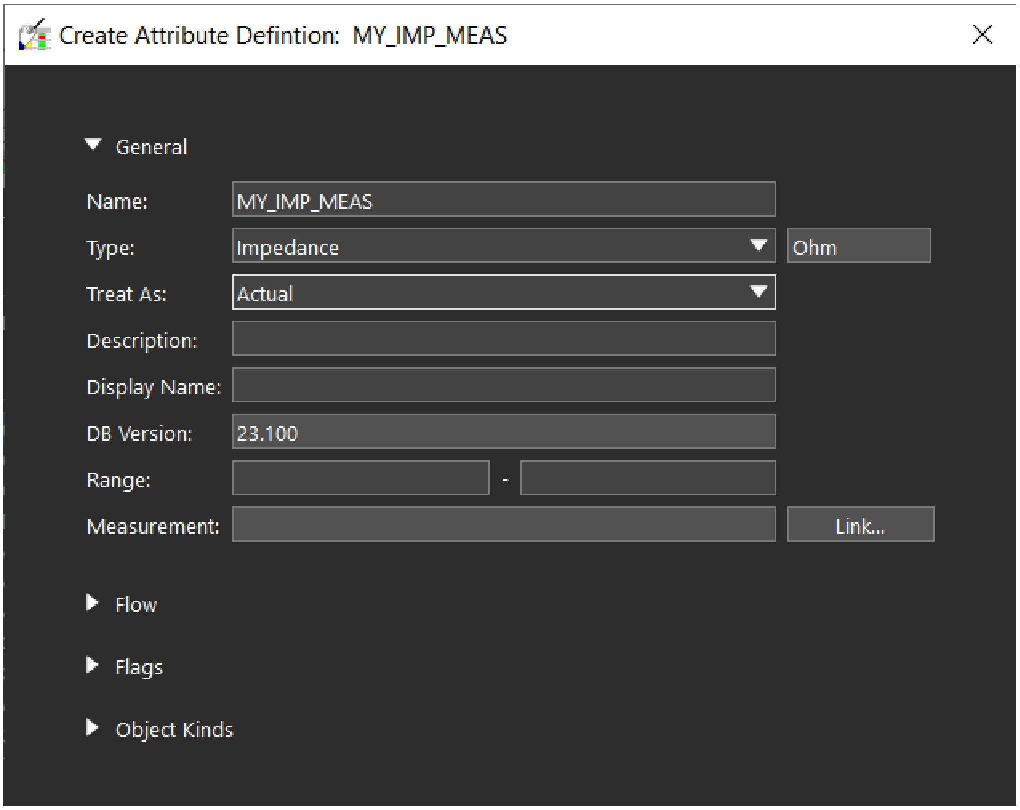
Name: MY_IMP_MEAS

Copying Option

□ Copy from attribute: DESCRIPTION

Figure 1-28 Create Attribute Dialog Box

7. Fill in the *Create Attribute Definition* dialog box as shown in Figure 1-29.



Create Attribute Defintion: MY_IMP_MEAS

▼ General

Name: MY_IMP_MEAS

Type: Impedance ▼ Ohm

Treat As: Actual ▼

Description:

Display Name:

DB Version: 23.100

Range: -

Measurement:

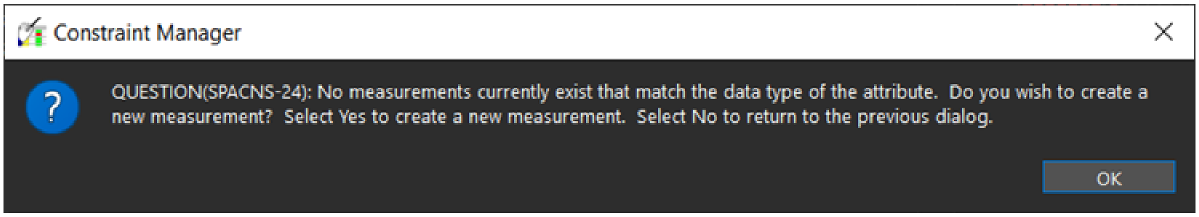
Figure 1-29 Create Attribute Definition Dialog Box

You are creating an *Actual* with the *Data Type* set as *Impedance*.

8. Click *Link* associated with *Measurement* field.

Previously, when you were creating a constraint, clicking *OK* launched a dialog box to select the desired associated measurement.

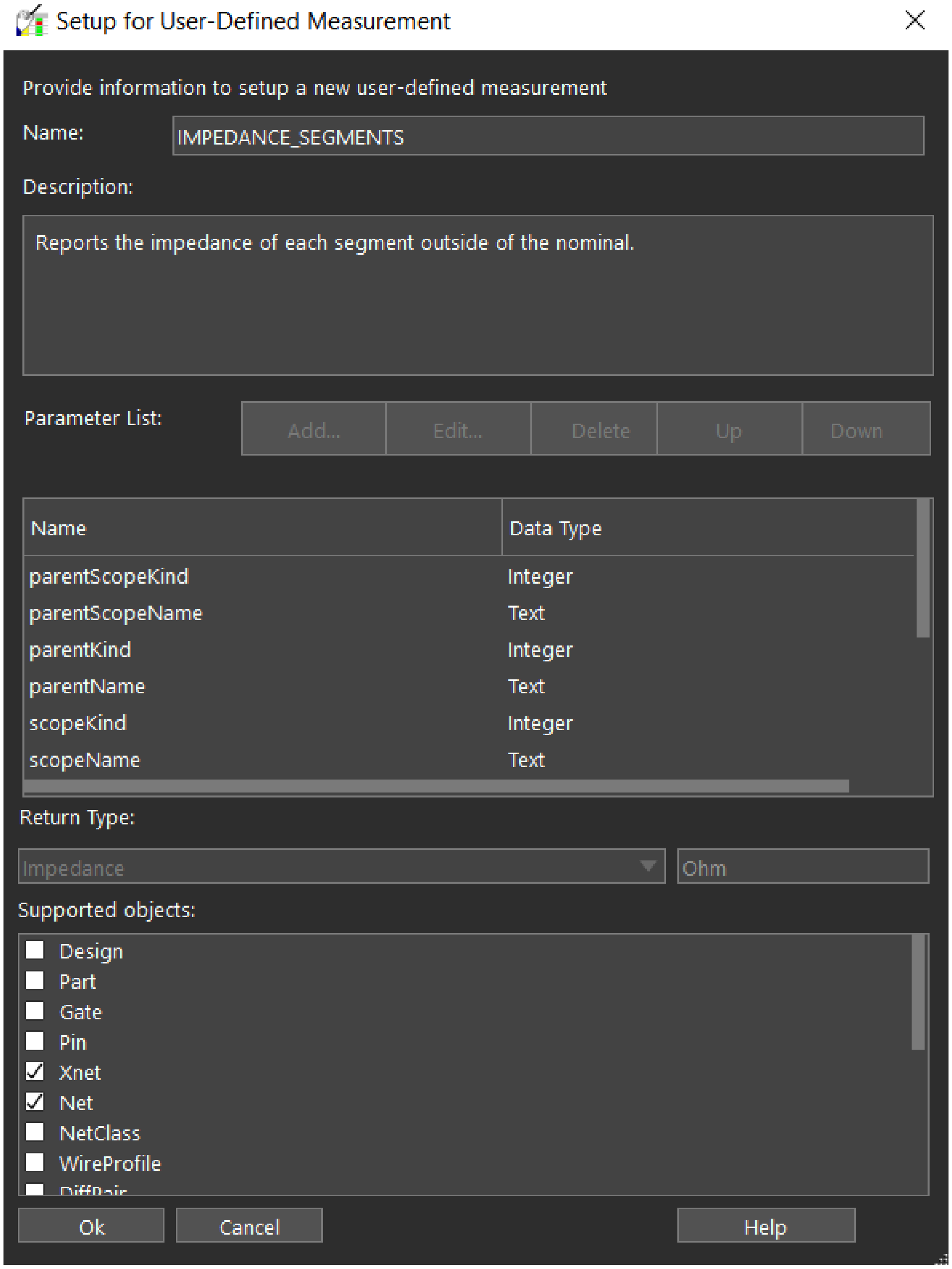
In this case, the list of Measurements is empty because there are no pre-defined Measurements with this Data Type (Impedance). A message is displayed prompting you to create a user defined measurement.



Constraint Manager

QUESTION(SPACNS-24): No measurements currently exist that match the data type of the attribute. Do you wish to create a new measurement? Select Yes to create a new measurement. Select No to return to the previous dialog.

9. Click *OK* in the message dialog box.



Setup for User-Defined Measurement

Provide information to setup a new user-defined measurement

Name: IMPEDANCE_SEGMENTS

Description:

Reports the impedance of each segment outside of the nominal.

Parameter List:

Name	Data Type
parentScopeKind	Integer
parentScopeName	Text
parentKind	Integer
parentName	Text
scopeKind	Integer
scopeName	Text

Return Type: Impedance Ohm

Supported objects:

- □ Design
- □ Part
- □ Gate
- □ Pin
- ☑ Xnet
- ☑ Net
- □ NetClass
- □ WireProfile
- □ DiffPair

Figure 1-30 Setup for User-Defined Measurement Dialog Box

10. Enter a *Name* and *Description*.

A similar dialog box is used for both User-Defined Measurement and User-Defined Predicates. In the case of User-Defined Measurement, the *Parameters List* is fixed so that portion of the dialog box has the buttons grayed out.

11. In the *Supported objects* section, make sure that only the following are selected:

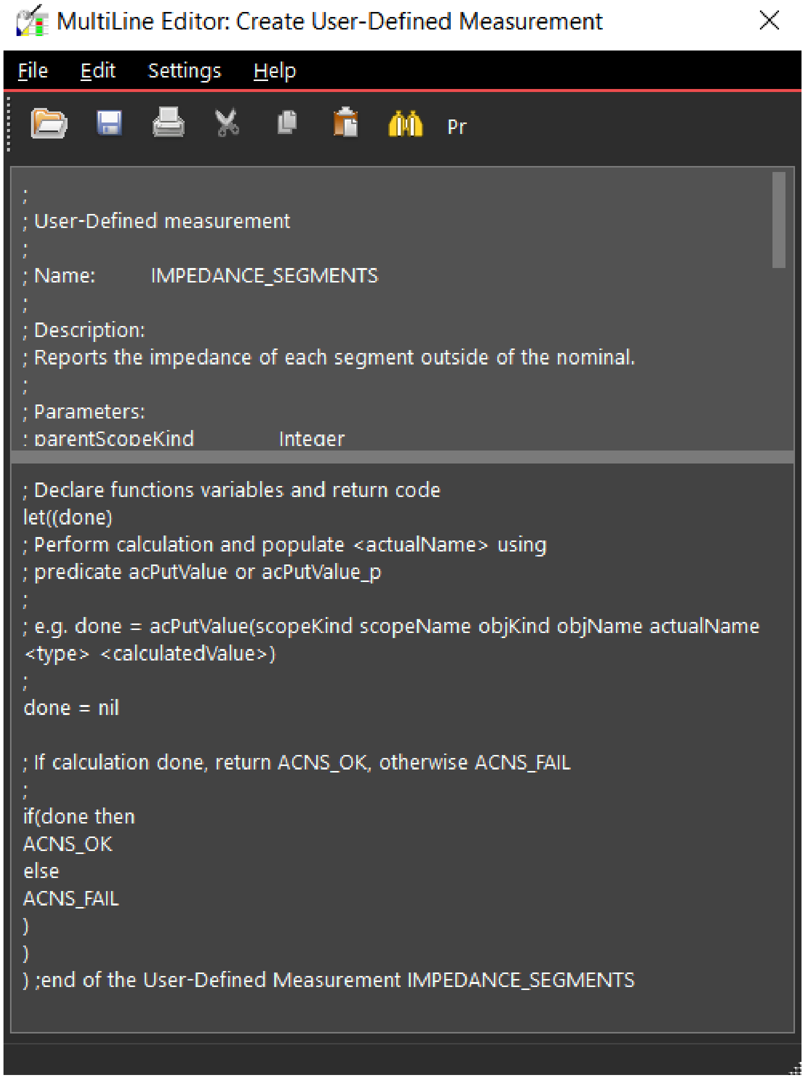
❑ Net

❑ Xnet

❑ Result

For Measurements, the *Supported objects* represent which kind of objects the measurement will analyze. Up to this point, you have ignored setting the *Objects* but it is a good practice to ensure that anything created in Constraint Manager has the correct *Objects* setting. This ensures that the appropriate (and desired) cells are enabled in Constraint Manager.

12. Click *OK* to display the MultiLine Editor specific to this Measurement.



```
;
; User-Defined measurement
;
; Name:        IMPEDANCE_SEGMENTS
;
; Description:
; Reports the impedance of each segment outside of the nominal.
;
; Parameters:
; parentScopeKind            Integer
```

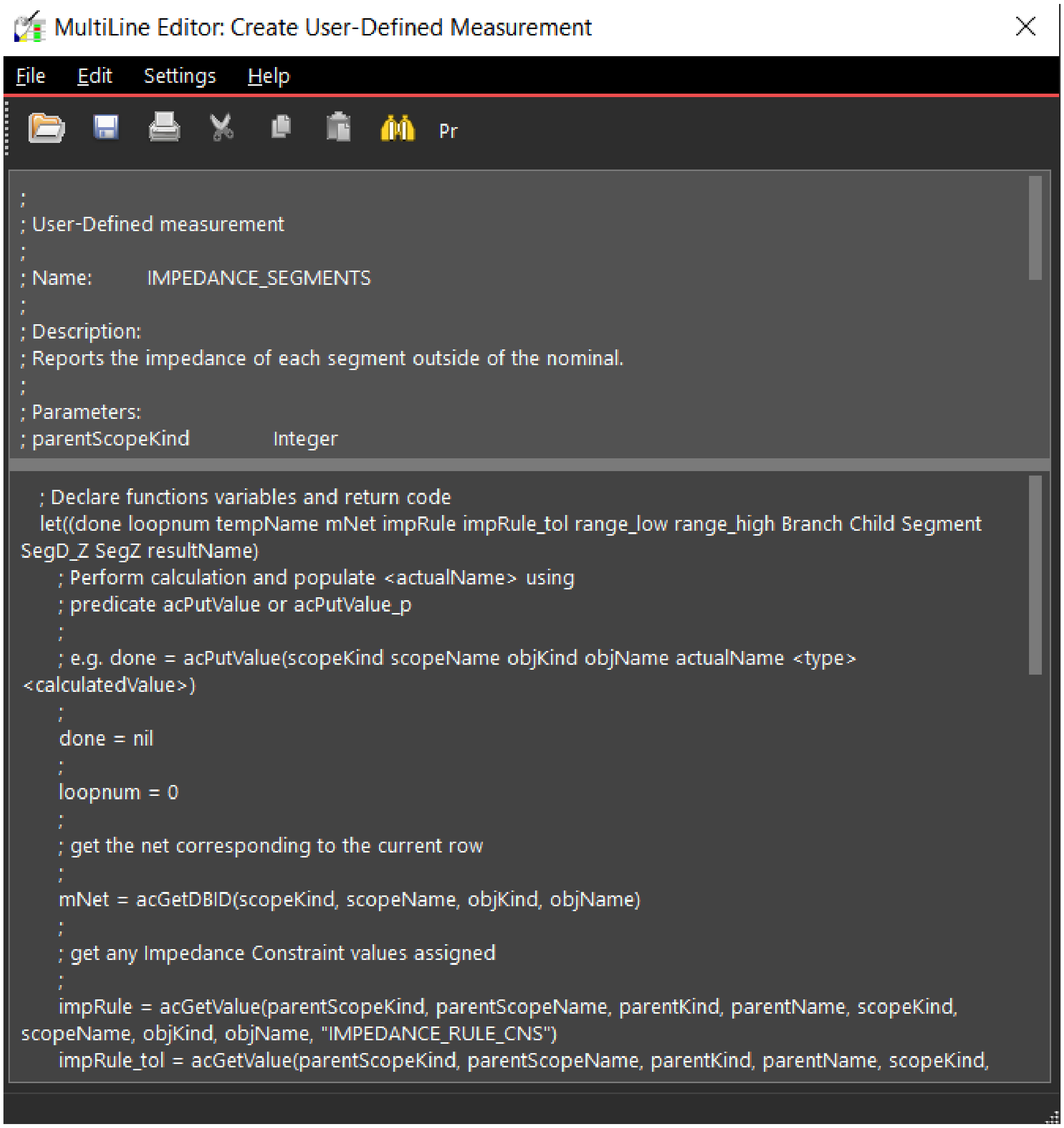
```
; Declare functions variables and return code
let((done)
; Perform calculation and populate <actualName> using
; predicate acPutValue or acPutValue_p
;
; e.g. done = acPutValue(scopeKind scopeName objKind objName actualName
<type> <calculatedValue>)
;
done = nil

; If calculation done, return ACNS_OK, otherwise ACNS_FAIL
;
if(done then
ACNS_OK
else
ACNS_FAIL
)
)
) ;end of the User-Defined Measurement IMPEDANCE_SEGMENTS
```

Figure 1-32 MultiLine Editor (Measurement)

13. Expand or scroll through the gray portion (read-only) at the top of the MultiLine Editor.

At the end of the gray section is a procedure call. Also notice that there is a template provided in the editing section of the dialog box. The template defines a variable *done* to indicate successful completion. In the comments section is a suggestion to assign done the value of the `acPutValue` predicate. Choose *File – Load from file* or click the associated icon and open the *impedance_meas.txt* file.



```
;
; User-Defined measurement
;
; Name:        IMPEDANCE_SEGMENTS
;
; Description:
; Reports the impedance of each segment outside of the nominal.
;
; Parameters:
; parentScopeKind          Integer
```

```
  ; Declare functions variables and return code
  let((done loopnum tempName mNet impRule impRule_tol range_low range_high Branch Child Segment
SegD_Z SegZ resultName)
    ; Perform calculation and populate <actualName> using
    ; predicate acPutValue or acPutValue_p
    ;
    ; e.g. done = acPutValue(scopeKind scopeName objKind objName actualName <type>
<calculatedValue>)
    ;
    done = nil
    ;
    loopnum = 0
    ;
    ; get the net corresponding to the current row
    ;
    mNet = acGetDBID(scopeKind, scopeName, objKind, objName)
    ;
    ; get any Impedance Constraint values assigned
    ;
    impRule = acGetValue(parentScopeKind, parentScopeName, parentKind, parentName, scopeKind,
scopeName, objKind, objName, "IMPEDANCE_RULE_CNS")
    impRule_tol = acGetValue(parentScopeKind, parentScopeName, parentKind, parentName, scopeKind,
```

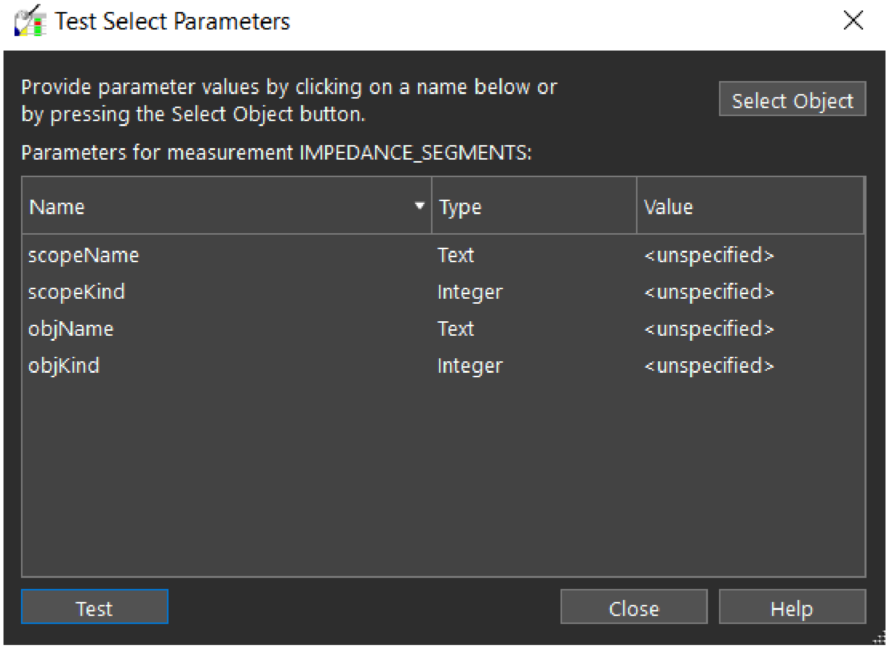
Figure 1-33 impedance_meas.txt File

In this example, notice:

- ❑ The list of variables added to the `let` statement
- ❑ The `acGetDBID` predicate; this gets DBID of an object to pass to AXL functions
- ❑ The use of an AXL function (`axlSegDelayAndZ0`)
- ❑ Conditional statements to filter the results

- The `acCreateResultName()` predicate used to define the name of a result object
- The `acAddResult()` predicate used to create the result object for the object being measured
- The `acPutValue()` predicate used to populate the *Actual* value on the result object

14. Choose *File – Test*.



Provide parameter values by clicking on a name below or by pressing the Select Object button.

Parameters for measurement IMPEDANCE_SEGMENTS:

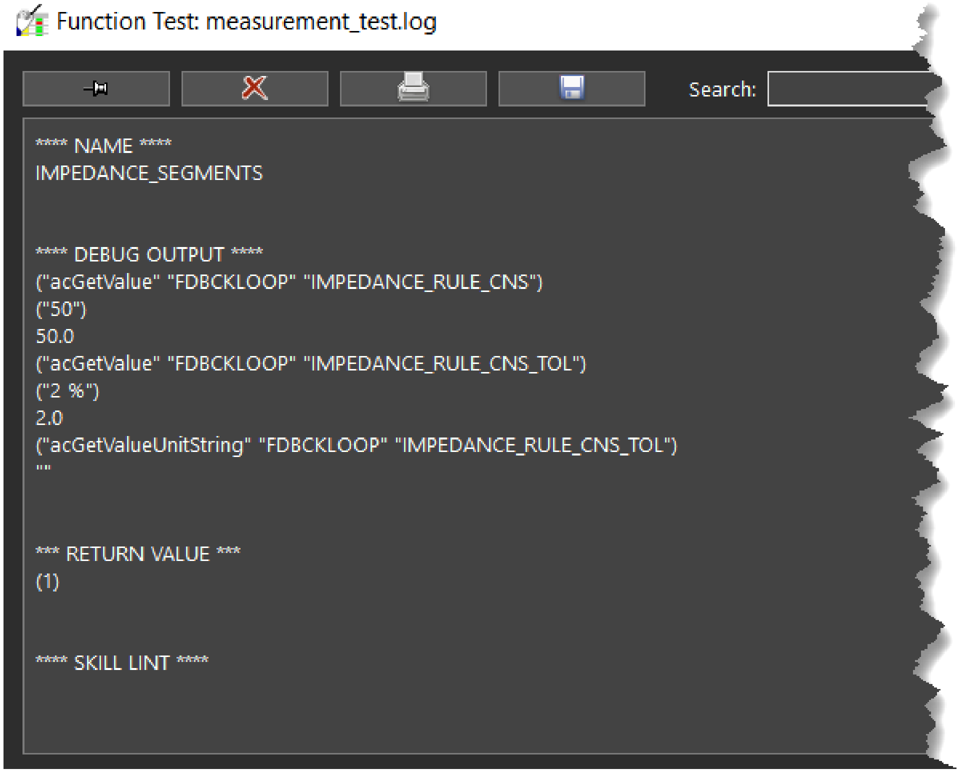
Name	Type	Value
scopeName	Text	<unspecified>
scopeKind	Integer	<unspecified>
objName	Text	<unspecified>
objKind	Integer	<unspecified>

Figure 1-34 Test Select Parameters

Note: The difference between the formula test environment discussed earlier and function test is that here the first step is a dialog that asks for values for the parameters.

15. Click *Select Object*.
16. Select *FDBCKLOOP* and click *OK* in the Select Object dialog box.
17. Click *Test*.

18. Compare your function test results with Figure 1-35.



```
**** NAME ****
IMPEDANCE_SEGMENTS


**** DEBUG OUTPUT ****
("acGetValue" "FDBCKLOOP" "IMPEDANCE_RULE_CNS")
("50")
50.0
("acGetValue" "FDBCKLOOP" "IMPEDANCE_RULE_CNS_TOL")
("2 %")
2.0
("acGetValueUnitString" "FDBCKLOOP" "IMPEDANCE_RULE_CNS_TOL")
""


*** RETURN VALUE ***
(1)


**** SKILL LINT ****
```

Figure 1-35 Function Test Results

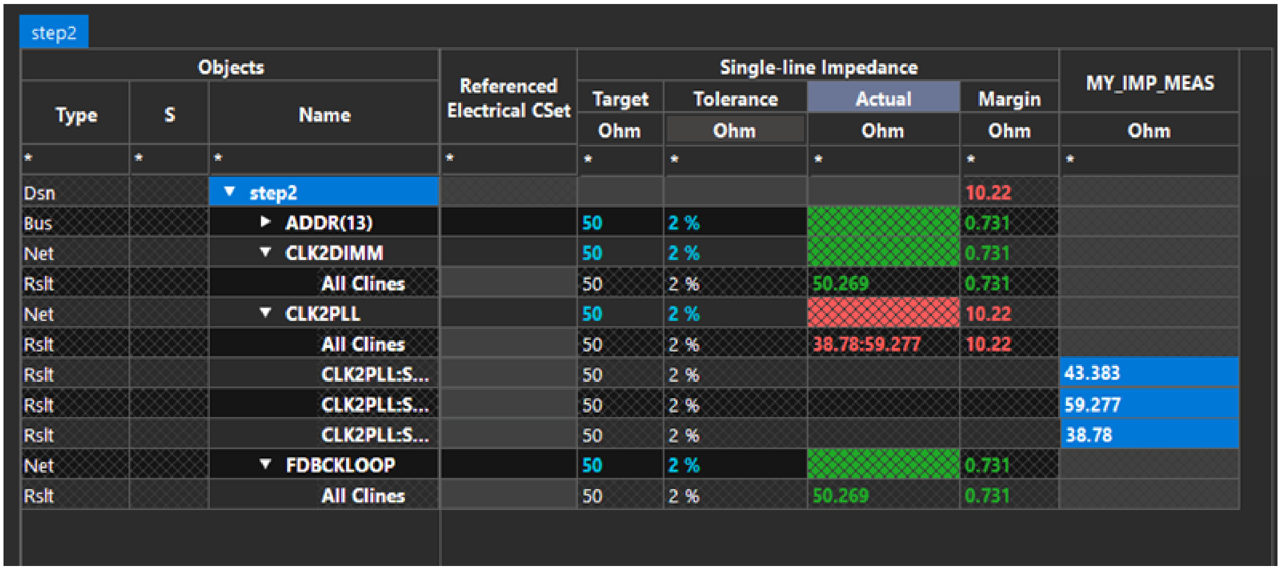
19. Close the window.

20. Save and close the MultiLine Editor.

21. Click *OK* in the *Select or Create Measurement* dialog box.

22. Click *OK* in the remaining two dialog boxes and any confirmer dialog box that appears.

23. Select *MY_IMP_MEAS* column header. Right-click on the design name and choose *Analyze* to generate results.



Objects			Referenced Electrical CSet	Single-line Impedance				MY_IMP_MEAS
Type	S	Name		Target	Tolerance	Actual	Margin	
				Ohm	Ohm	Ohm	Ohm	Ohm
*	*	*	*	*	*	*	*	*
Dsn		▼ step2					10.22	
Bus		▶ ADDR(13)		50	2 %	🟩	0.731	
Net		▼ CLK2DIMM		50	2 %	🟩	0.731	
Rslt		All Clines		50	2 %	50.269	0.731	
Net		▼ CLK2PLL		50	2 %	🟥	10.22	
Rslt		All Clines		50	2 %	38.78:59.277	10.22	
Rslt		CLK2PLL:S...		50	2 %			43.383
Rslt		CLK2PLL:S...		50	2 %			59.277
Rslt		CLK2PLL:S...		50	2 %			38.78
Net		▼ FDBCKLOOP		50	2 %	🟩	0.731	
Rslt		All Clines		50	2 %	50.269	0.731	

Figure 1-36 Analysis of step2.brd Design

Even though you analyzed the entire design, you got results only for the *CLK2PLL* net, which is expected since it is the only net with errors. Notice that in fact there are three segments that are outside the *Target / Tolerance* range.

Although what you created and added is only an *Actual and is being used as a standalone measurement*, it could also be used as an Actual in a user-defined constraint.