

Virtuoso EDIF 200 Reader and Writer SKILL Reference

**Product Version IC23.1
June 2023**

© 2023 Cadence Design Systems, Inc. All rights reserved.
Printed in the United States of America.

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

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.

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.

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.

Patents: The Cadence Products covered in this manual are protected by U.S. Patents 5,790,436; 5,812,431; 5,859,785; 5,949,992; 6,493,849; 6,278,964; 6,300,765; 6,304,097; 6,414,498; 6,560,755; 6,618,837; 6,693,439; 6,826,736; 6,851,097; 6,711,725; 6,832,358; 6,874,133; 6,918,102; 6,954,908; 6,957,400; 7,003,745; 7,003,749.

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 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

<u>1</u>	
<u>Customizing EDIF 200 In</u>	5
<u>Licensing Requirements</u>	5
<u>edifinDisplay</u>	6
<u>edifLayerNumMap</u>	7
<u>edifinMakeRenameString</u>	9
<u>ediFinishStatus</u>	11

Virtuoso EDIF 200 Reader SKILL Reference

Customizing EDIF 200 In

You can customize EDIF 200 In output files by creating a SKILL file. A SKILL file contains SKILL procedures that specify changes to object properties in the source design.

This section lists the SKILL procedures you can add to a SKILL file and provides an example of each procedure.

You use the *User-Defined SKILL File* field on the EDIF 200 In form to specify the name of the SKILL file. Specify the full path if it is not in your run directory.

When you start EDIF 200 In, the software automatically looks for the SKILL file in the run directory or in the directory you specify in the *User-Defined SKILL File* field. When the SKILL file is found, the software runs the procedures and modifies the property information that is written to the EDIF 200 In output files.

A SKILL procedure returns a value of `t` when it runs successfully and a value of `nil` when it fails. When a procedure fails, EDIF 200 In processes the property, if possible, as the property is described in the source EDIF file.

See the [Cadence SKILL Language User Guide](#) and the [Cadence SKILL Language Reference](#) for information about writing SKILL procedures.

Licensing Requirements

For information on licensing in the Virtuoso Studio Design Environment, see [Virtuoso Software Licensing and Configuration Guide](#).

edifinDisplay

```
edifinDisplay(  
    t_edifFormName  
)  
=> t / nil
```

Description

Displays the EDIF 200 In form.

Arguments

<i>t_edifFormName</i>	Specifies the Edif200 Import form name.
-----------------------	-----------------------------------------

Value Returned

<i>t</i>	Indicates that the command succeeded.
<i>nil</i>	Indicates that the command failed.

Example

```
edifinDisplay(transEdifInForm)
```

edifLayerNumMap

```
edifLayerNumMap(  
    t_figureGroup  
)  
=> list( t_layerName t_purposeName ) / nil
```

Description

Maps the name of `figureGroup` specified in the input EDIF file to the layer-purpose pairs in the technology file.

The technology file is used when libraries are created by EDIF 200 In. The layers defined by `edifLayerNumMap` overwrite the default layers. The procedure returns a list that contains two strings that specify the layer and purpose names.

Arguments

<code>t_figureGroup</code>	A <code>figureGroup</code> name in the EDIF input file.
----------------------------	---------------------------------------------------------

Value Returned

<code>t_layerName</code>	Returns <code>layerName</code> , which maps to the specified <code>figureGroup</code> .
<code>t_purposeName</code>	Returns <code>purposeName</code> , which maps to the specified <code>figureGroup</code> .
<code>nil</code>	Returns <code>nil</code> if mapping is not possible.

Skeletal Example

```
procedure( edifLayerNumMap( figureGroup "t" )  
    prog((flag)  
        flag = case( figureGroup  
            ("arg1" return( list( "arg2" "arg3" )))  
        ) ;end case  
        if(!flag then  
            return( list( "arg2" "arg3" ))  
        );end if  
    );prog  
);procedure
```

<code>"arg1"</code>	Specifies the <code>figureGroup</code> name in the input EDIF file.
---------------------	---------------------------------------------------------------------

Virtuoso EDIF 200 Reader SKILL Reference

Customizing EDIF 200 In

"arg2 "	Specifies the DFII layer to which the figureGroup is mapped.
"arg3 "	Specifies the DFII layer-purpose to which the figureGroup is mapped.

Example

```
procedure( edifLayerNumMap(figureGroup "t")
  prog((flag)
    flag = case( figureGroup
      ("DEVICE_BODY_FGP" return( list( "device" "drawing" )))
      ("NET_FGP" return( list( "wire" "drawing" )))
      ("PIN_FGP" return( list( "pin" "drawing" )))
      ("PIN_BODY_FGP" return( list( "pin" "drawing" )))
    );end case
    if(!flag then
      return( list( "device" "drawing" ))
    );end if
  );prog
);procedure
```

In this example, any figureGroup names that are not among those listed in the case statement are mapped to the layer "device" with the "drawing" purpose. The default layer name for port is "pin". The default layer name for net is "wire". The default layer name for net label is "wire" and the layer-purpose is "label".

edifinMakeRenameString

```
edifinMakeRenameString(  
    t_inputString  
)  
=> l_outputString / nil
```

Description

Creates a special string when the string in the EDIF 200 In input file is an illegal name.

The function takes the illegal name string from the EDIF file as an argument and returns a list value that contains the new rename string. You can use `nil` as the `returnString` value. In this case, a `nil` value is returned, which indicates that no new string was provided and the original characters are deleted.

Arguments

<i>t_inputString</i>	An illegal name in the EDIF input file.
----------------------	-----------------------------------------

Value Returned

<i>l_outputString</i>	Indicates the list that contains the new rename string.
<i>nil</i>	Indicates no new string was provided.

Skeletal Example

```
procedure( edifinMakeRenameString( inputString "t" )  
    prog( (returnString)  
        case( inputString  
            ( "arg1" returnString = "arg2" ))  
        return( list( returnString ))  
    );prog  
);procedure
```

"arg1"	Specifies the string in the EDIF file to map.
"arg2"	Specifies the translated string in the DFII file.

Virtuoso EDIF 200 Reader SKILL Reference

Customizing EDIF 200 In

To map names to a DFII database, you can use any alphanumeric character and any of the following special characters:

_ (underscore)	+ (plus)	- (hyphen)	: (colon)
<> (angle brackets)	{ } (braces)	[] (square brackets)	() (parentheses)

If you use an invalid character, EDIF 200 In ignores the name and uses the current EDIF name.

Note: You can use this mapping convention to map the `instanceName`, `portName`, and `netName` constructs. However, when you rename instance names, the `charMapForInstName` function overrides this `edifinMakeRenameString` function.

Example

```
procedure( edifinMakeRenameString( inputString )
    prog( (returnString)
        case( inputString
            ( "C4" returnString = "C<4>" )
        ); case changes name "C4" into "C<4>"
        return( list( returnString)
    );prog
);procedure
```

The `inputString` is the name of the string in the EDIF file to map. The `returnString` is the name of the DFII database name that is created.

ediFinishStatus

```
ediFinishStatus(  
    x_returnCode  
)
```

Description

Provides an exit status number when EDIF 200 In is done.

Arguments

- | | |
|---------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>x_returnCode</i> | <ul style="list-style-type: none">■ 0—means there were no errors and no warnings.■ -1—means there were errors and warnings.■ 1—means there were warnings but no errors. |
|---------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Example

```
procedure( ediFinishStatus( returnCode "x")  
returnCode = 0 if there are no errors and no warnings  
returnCode = -1 if there are errors and warnings  
returnCode = 1 is there are warnings and no errors  
)
```

Virtuoso EDIF 200 Reader SKILL Reference

Customizing EDIF 200 In
