

SANDIA REPORT

SAND2021-5536
Printed May 2021



Xyce™ XDM Netlist Translator User Guide, Version 2.3

Gary J. Templet, Garrick Ng, Richard L. Schiek, Peter E. Sholander, Jason C. Verley

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185
Livermore, California 94550

Issued by Sandia National Laboratories, operated for the United States Department of Energy by National Technology & Engineering Solutions of Sandia, LLC.

NOTICE: This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government, nor any agency thereof, nor any of their employees, nor any of their contractors, subcontractors, or their employees, make any warranty, express or implied, or assume any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represent that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government, any agency thereof, or any of their contractors or subcontractors. The views and opinions expressed herein do not necessarily state or reflect those of the United States Government, any agency thereof, or any of their contractors.

Printed in the United States of America. This report has been reproduced directly from the best available copy.

Available to DOE and DOE contractors from

U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865) 576-8401
Facsimile: (865) 576-5728
E-Mail: reports@osti.gov
Online ordering: <http://www.osti.gov/scitech>

Available to the public from

U.S. Department of Commerce
National Technical Information Service
5301 Shawnee Road
Alexandria, VA 22312

Telephone: (800) 553-6847
Facsimile: (703) 605-6900
E-Mail: orders@ntis.gov
Online order: <https://classic.ntis.gov/help/order-methods>



ABSTRACT

This manual describes the installation and use of the **Xyce™** XDM Netlist Translator. XDM simplifies the translation of netlists generated by commercial circuit simulator tools into Xyce-compatible netlists. XDM currently supports translation from PSpice, HSPICE, and Spectre netlists into **Xyce™** netlists.

CONTENTS

1. Introduction	11
2. XDM Installation	12
2.1. Windows	12
2.2. Linux and OS X	12
3. Translating Netlists with XDM	15
3.1. Startup	15
3.1.1. Windows	15
3.1.2. Linux/OS X	15
3.2. Running XDM	15
4. PSpice Translations: Overview and Known Issues	19
5. HSPICE Translations: Overview and Known Issues	21
5.1. Commands known not to translate	22
5.2. Xyce <code>-hspice-ext</code> Command Line Option	23
6. Spectre Translations: Overview and Known Issues	25
6.1. Commands known not to translate	26
7. Translation of PDK Model Libraries	27
8. Release Notes	29
8.1. XDM 2.3	29
8.1.1. General	29
8.1.2. PSpice	29
8.1.3. HSPICE	29
8.1.4. Spectre	29
References	31
Appendices	32
A. Third Party Licenses	32

LIST OF FIGURES

Figure 3-1. Example of an XDM Warning message.	17
Figure 3-2. Example of an XDM Error message.	17

LIST OF TABLES

Table 5-1. <code>-hspice-ext</code> Command Line Option	23
---	----

1. INTRODUCTION

The Xyce XDM Netlist Translator is a software solution developed by Sandia National Laboratories for translating SPICE netlists into a format readable by Xyce, Sandia's internally developed SPICE-compatible simulation tool.

XDM (Xyce Data Model) can translate netlists written in PSpice, HSPICE, and Spectre formats. XDM's PSpice translation capabilities are the most mature and can be considered production quality. Similarly, XDM's HSPICE translation capabilities, introduced in the 2.0 release, are approaching production quality. Xyce has the command line option `-hspice-ext` to handle some cases where XDM doesn't translate HSPICE syntax. PSpice and Spectre translations require no additional Xyce command line options. Spectre translations are new in version 2.3 and improving with each new release. For all netlist languages XDM supports, and for XDM itself, feedback from users is always appreciated. XDM users can send feedback, bug reports, and feature requests to, in order of preference:

- Xyce's Google Group: <https://groups.google.com/g/xyce-users/>
- XDM's Github Issue board: <https://github.com/Xyce/XDM>
- Xyce's Contact Page: https://xyce.sandia.gov/contact_us.html
- Xyce Email: xyce@sandia.gov

Sandia supplies binaries for RHEL7, OS X, and Windows that have the Boost and Python3 dependencies built in. The XDM executable is named "xdm_bdl", short for "xdm_bundle", and was chosen over "xdm" to avoid name clashes with the X11 "xdm" tool. "xdm_bdl" is a standalone C++ and Python program. New XDM releases are delivered as part of each new Xyce release.

XDM can be built on a number of platforms including Linux, OS X, and Windows. Building XDM requires the Boost Python libraries built with Python 3 and "PyInstaller". Build instructions for XDM are included in the "README.md" file found in the XDM source available at <https://github.com/Xyce/XDM>.

2. XDM INSTALLATION

The installation procedure for the XDM binaries is substantially the same on Windows, Linux, and OS X (Darwin) systems with changes for system-dependent commands. For each of these operating systems XDM is delivered in binary form as a “zip” file:

- xdm-2.3.0-win64.zip
- xdm-2.3.0-Linux.zip
- xdm-2.3.0-Darwin.zip.

To install XDM unzip the file for the target system and then copy or move the directory it contains to the desired installation location. It is recommended to install XDM in the same location as Xyce. This helps users find XDM when running Xyce, and serves as a place to store different release versions of XDM, though the latest XDM release is always recommended. Finally, the XDM executable `xdm_bdl` is a standalone executable invoked from the command line. The path to `xdm_bdl` can be added to the “PATH” environment variable to avoid entering the full path each time `xdm_bdl` is invoked.

2.1. Windows

The Windows installation steps are:

- Open `xdm-2.3.0-win64.zip` with a double click
- Copy the unzipped folder `xdm-2.3.0-win64` to the location of your choice. The directory in which Xyce was installed is a good choice. By default the Xyce package is installed in the `C:\Program Files\` folder.
- Inside the `xdm-2.3.0-win64\bin` folder, a batch file, called `xdm_console`, is provided to automatically open a command window that has the XDM binary `xdm_bdl` in the path. This enables “`xdm_bdl`” to work without typing its full path for each invocation.
- For convenience, users can create a link to the `xdm_console` file on the desktop by right clicking on the file and selecting “Create shortcut.”

2.2. Linux and OS X

The Linux and OS X installation steps are:

- Open/Unzip the `xdm-2.3.0-Linux.zip` or `xdm-2.3.0-Darwin.zip` using the “unzip” or “tar” utilities found on Unix-like systems.
- Copy the unzipped folder `xdm-2.3.0-Linux` or `xdm-2.3.0-Darwin` to the location of your choice. The Xyce installation location is a good choice. By default this is `/usr/local` on Unix-like systems.

- Unlike Windows environments there is no batch file to open a shell window with the `xdm_bdl` automatically loaded into the `PATH` variable. Instead users can edit their `PATH` environment variable in a couple ways:

- Manually with command below for the current “Bash” and “Z Shell” windows.

```
export PATH=$PATH:/usr/local/xdm-2.3.0-Linux/bin
```

```
export PATH=$PATH:/usr/local/xdm-2.3.0-Darwin/bin
```

- Automatically on shell startup by adding the appropriate command above for `.bashrc` or `.zshrc`.

3. TRANSLATING NETLISTS WITH XDM

3.1. Startup

3.1.1. Windows

If you created a link to the `xdm_console` batch file (as described in Chapter 2), then double-click the file. Otherwise, start the “Command Prompt”. The remaining instructions assume the XDM binary is in your path.

3.1.2. Linux/OS X

Start a terminal shell. It is assumed that you have the XDM binary in your path.

3.2. Running XDM

An example XDM invocation for translating a file is

```
xdm_bdl -s hspice -d out -o xyce --auto test_circuit.sp
```

where

- `-s hspice` designates an HSPICE input format
- `-d out` designates writing output to a directory called, “out”
- `-o xyce` designates the Xyce output format
- `--auto` specifies that XDM should automatically translate any `.INC/.LIB` files
- `test_circuit.sp` is the input circuit to be translated

To see a full description of XDM’s flags, type `xdm_bdl -h` on the command line:

```
$ xdm_bdl -h
usage: xdm_bdl [-h] [-s [{hspice, tspice, pspice, spectre, xyce}]] [-d [DIR_OUT]]
              [-o [{xyce}]] [--auto] [--eval] [-l {DEBUG, INFO, WARN, ERROR}]
              [-q {R, C, D, L, X, Q, ALL}] [--license]
              input_file
```

xdm 2.0.0: Translates input netlist file by creating a new netlist file of a different netlist file format. The translated input file (of the same name) is written into the specified output directory - if you used the same directory as the input netlist file, the original file will be overwritten. xdm also

supports a device query interface for the SAW environment.

positional arguments:

input_file The input netlist file

optional arguments:

-h, --help show this help message and exit
-s [{hspice, tspice, pspice, spectre, xyce}],
 --source_file_format [{hspice, tspice, pspice, spectre, xyce}]
 The source/input netlist file format (default: pspice)
-d [DIR_OUT], --dir_out [DIR_OUT]
 The output directory (default: default_dir)
-o [{xyce}], --output_file_format [{xyce}]
 The output netlist file format (default: xyce)
--auto Automatically translate include and library files
 (default: False)
--eval Evaluate functions during translation (default: False)
-l {DEBUG, INFO, WARN, ERROR}, --logging {DEBUG, INFO, WARN, ERROR}
 Control the level of screen logging output: WARN is
 quiet - only ERROR and WARN level messages will be
 sent to the screen (default: WARN)
-q {R, C, D, L, X, Q, ALL}, --query_device {R, C, D, L, X, Q, ALL}
 Query for a device type of interest within the SAW
 environment (default: None)
--license Display the license for this version of XDM (default:
 None)

Note that, while several input netlist formats are listed, only HSPICE, PSpice, and Spectre have functional support.

After XDM runs, the translated circuit file and its associated library files in the above example should be in the “out” directory. To run the circuit file in Xyce, the command:

```
Xyce test_circuit.sp -hspice-ext all
```

should be used to resolve any remaining translation issues not handled by XDM.

When XDM encounters something it doesn’t understand, it will either produce a warning message, and continue processing, or print an error message and stop processing. Typically, a warning message is produced when a specific line cannot be translated. In these cases, the user can choose to ignore the issue, or may be able to find an equivalent Xyce syntax by hand. An example warning message is shown in Figure 3-1. The line, in the input file, that caused the warning message is then typically left as a comment line in the translated netlist.

Error messages occur in cases where XDM cannot continue processing, such as a missing file, or non-ASCII characters appearing in the netlist. XDM will give as much information as possible for the source of the error. An example warning message is shown in Figure 3-2.

PSpice file name and line #

Text of commented-out PSpice line

```
xdm_bdl.exe 1.4.0 (last changed on 2017-08-04 14:46:44) at
C:\Users\Desktop\xdm-1.4.0-win64\bin\xdm_bdl

is translating the file:

    'pwl_sources.cir.pspice' (input format = pspice)
        using xml definition C:\Users\AppData\Local\Temp\_ME113~2\pspice.xml
    => and is creating the translated files under the directory 'pwl_sources.cir-translated' (output format = xyce)
        using xml definition C:\Users\AppData\Local\Temp\_ME113~2\xyce.xml

Original calling command for this run was:

    xdm_bdl.exe '-s' 'pspice' '-d' 'pwl_sources.cir-translated' '-o' 'xyce' 'pwl_sources.cir.pspice'

    08/07/2017 10:00:55 AM WARNING: In file:"SCHEMATIC1.net" at line:[32, 33]. * V_V5 1 0 PWL REPEAT FOREVER
(0,0) (5e-4,1) (1e-3,0) ENDREPEAT; PSpice Parser Retained (as a comment). Continuing.

=== xdm execution complete:

Total critical issues reported      = 0:
Total      errors reported          = 0:
Total      warnings reported        = 1:
Total      information messages reported = 5:

SUCCESS: xdm completion status flag = 0:
```

"0" = success

Figure 3-1. Example of an XDM Warning message.

PSpice file name

Error message gives basic info on cause of fatal error.

```
xdm_bdl.exe 1.4.0 (last changed on 2017-08-04 14:46:44) at
C:\Users\Desktop\xdm-1.4.0-win64\bin\xdm_bdl

is translating the file:

    'example1.cir' (input format = pspice)
        using xml definition C:\Users\AppData\Local\Temp\_ME113~2\pspice.xml
    => and is creating the translated files under the directory 'example1.cir-translated' (output format = xyce)
        using xml definition C:\Users\AppData\Local\Temp\_ME113~2\xyce.xml

Original calling command for this run was:

    xdm_bdl.exe '-s' 'pspice' '-d' 'example1.cir-translated' '-o' 'xyce' 'example1.cir'

    08/07/2017 01:09:02 PM ERROR: File: C:\XDM_1.4.0_Windows_Test\Bogo.lib was not found. Please locate this file and try again.
```

Figure 3-2. Example of an XDM Error message.

4. PSPICE TRANSLATIONS: OVERVIEW AND KNOWN ISSUES

Libraries and Models XDM does not throw a fatal error or emit a warning if a `.MODEL` statement is missing for a model that requires one. As a result, Xyce will emit an error message during netlist parsing.

Unsupported Device Groups XDM does not yet support the following groups of devices, because of differences in syntax and parameters:

- Digital device models
- T devices (lossy/lossless transmission lines)

Unsupported PSpice Syntax Some PSpice options and commands do not have a legal Xyce translation. The following are always commented out by XDM, but should not affect the Xyce simulation:

- `.AUTOCONVERGE`
- `.OPTIONS ADVCONV`
- `.OPTIONS CHGTOL`
- `.OPTIONS ITL2`
- `.OPTIONS VNTOL`

Undocumented PSpice Syntax Some PSpice syntax is undocumented in the PSpice guides. The following are known examples (for device instance lines and `.MODEL` statements) that aren't supported by XDM:

- Missing commas in TC specifications
TC=0.1 0.1
vs.
TC=0.1, 0.1
 - The latter is what is documented. The former will be commented out by XDM.
- Missing parameter values
TC= or BV= without a value
- Extra matched set of parentheses
`.MODEL D1N3940 D((BV=600 DEV=1) IS=4E-10 RS=.105 N=1.48 CJO=1.95E-11)`
- Unmatched right/left parentheses in `.MODEL` statements
`.MODEL D1N3940 D((BV=600 DEV=1%) IS=4E-10 RS=.105 N=1.48 CJO=1.95E-11)`

Piecewise Linear (PWL) Sources XDM is able to handle these PSpice forms of PWL:

- Documented syntax in the PSpice Reference Guide
`V_V1 1 0 PWL (1e-3,0.5) (2e-3,1) (3e-3,1) (4e-3,0.5)`
- Undocumented syntax in the PSpice Reference Guide
`V_V1 1 0 PWL 1e-3 0.5 2e-3 1 3e-3 1 4e-3 0.5`
`V_V2 2 0 PWL (1e-3, 0.5, 2e-3, 1, 3e-3, 1, 4e-3, 0.5)`
`V_V3 3 0 PWL (1e-3 0.5 2e-3 1 3e-3 1 4e-3 0.5)`
- Using a file
`V_FILE1 5 0 PWL FILE "pwlFile1.txt"`

The following PSpice forms of PWL will be commented out by XDM, since they don't have a legal (or straightforward) translation in Xyce:

- Some instances of REPEAT
See Section 6.1.12, "*Piecewise Linear Sources*" of the Xyce Reference Guide [1] for more details on how to manually translate these PSpice PWL instance lines.
- Anything using TIME_SCALE_FACTOR or VALUE_SCALE_FACTOR

Controlled sources While the documented syntax of the POLY form works for the F and G sources, the subcircuit syntax produced by OrCAD capture is not always consistent with the documentation.

.PROBE and .PROBE64 Some of the wildcard forms used by .PROBE and .PROBE64 are not supported:

- /V () all voltages
- -/V () no voltages
- -/V (X) all voltages except internal sub-circuit voltages

5. HSPICE TRANSLATIONS: OVERVIEW AND KNOWN ISSUES

HSPICE .OPTIONS With one exception, HSPICE .OPTIONS lines will be commented out in the translated Xyce netlist for the following reasons:

- There may not be a direct translation of HSPICE options into Xyce options.
- Even options with the same or similar names in HSPICE and Xyce may not have the same functionality.
- The exception is the TNOM option in HSPICE, which will be translated in the Xyce netlist.

Instance parameters Issues to keep in mind with regards to translations of instance parameters:

- Instance parameters of devices that exist in HSPICE but not in Xyce will be automatically removed. E.g., DTEMP will be removed for R, L and C devices. *Watch for this in the on-screen warnings.*
- Care needs to be used when deciding whether to translate with the `-auto` option. If a device is an instantiation of a model declared in another file, and auto translate is turned off, that device's instance parameters will default to parameters using a default level (typically `level=1`). If the different model levels have different allowed instance parameters, this may result in parameters being removed.

Wildcards Wildcards in .PRINT statements are generally illegal in Xyce, and will be commented out of the resulting netlist by XDM. The only exception is `V(*)`, which is allowed and will be translated by XDM.

Multiplicity (M Factor) In HSPICE, the “multiplicity” (or “M Factor”) can be used to specify multiple netlist devices in parallel via a single instance line. In Xyce, the terms “multiplicity factor” and “multiplier” are used to describe that same concept.

At present, the multiplicity factor (M parameter) is only supported in Xyce by the R, L, C and MOSFET device models, and some BJT device models (VBIC 1.3 and MEXTRAM). It is not supported for the X device (subcircuits).

Solution-Dependent Resistors In HSPICE, it is legal to have the resistance value for an R device depend on a solution variable. An example is as follows, where P1 is a parameter defined elsewhere in the netlist:

```
R1      1      2      'P1*v(in,out)'
```

Xyce does not support solution-dependent resistors, but the equivalent behavior can be specified using the Xyce Nonlinear Dependent Source (B device). XDM will automatically make this translation. However, it is important to note that, if the resistor value becomes zero, the expression becomes undefined, and overflows will occur. For example, the above resistor would be translated to the following:

```
B1      1      2      I={ (V(2)-V(1) / (P1*v(in,out) ) }'
```

If, during the simulation the pin voltages are the same ($in=out$), then $v(in,out)=0$, and the expression results in an infinite current.

AGAUSS and AUNIF The AGAUSS and GAUSS functions are defined both in HSPICE and Xyce to handle Gaussian distributions. For uniform distributions, HSPICE then uses the AUNIF and UNIF functions, while Xyce uses the RAND function. The Xyce definitions are given in the “Expressions” section of the Xyce Reference Guide [1]. The HSPICE and Xyce versions of AGAUSS, GAUSS, and their respective AUNIF, UNIF and RAND functions, are not yet fully compatible:

- In HSPICE, if Monte Carlo (MC) sampling is *not* turned on, then the AGAUSS and GAUSS functions just return the mean of the distribution. If MC sampling is turned on, then HSPICE will randomly sample the specified distributions.
- The default Xyce behavior for the AGAUSS and GAUSS functions is to return a random number from the specified Gaussian distribution. The “-hspice-ext random” command line option will make them return the mean (see Section 5.2).

5.1. Commands known not to translate

The following command lines, found in HSPICE, are not directly supported in Xyce:

- .ALTER
- .TEMP
- .IF, .ELSEIF, .ELSE and .ENDIF

In addition, any usages of DTEMP for subcircuits will likely need to be replaced with explicit parameter values, especially for DTEMP not equal to 0.

.NODESET The Xyce .NODESET command uses a different strategy than either SPICE3F5 or HSPICE. So, the Xyce behavior may differ from that provided by .NODESET and .OPTION DCHOLD in HSPICE. In addition, Xyce does not allow the use of “wildcards” in .NODESET (or .IC) statements. The “.NODESET (Approximate Initial Condition, Bias point)” section of the Xyce Reference Guide [1] gives more details on the Xyce implementation.

.OPTION MACMOD The various HSPICE MACMOD options allow HSPICE to search for subcircuit definitions or Verilog-A definitions in place of model references and/or vice-versa (depending on the specification). Xyce does not support this option at this time.

Verilog-A Support Xyce does have the capability to dynamically link in Verilog-A models. However, that capability is limited and not HSPICE compatible. In particular, it is not possible to insert Verilog-A models into Xyce via the netlist alone. So, Xyce does not support the HSPICE `.HDL` command.

Multiple .END Statements The following netlist is legal in HSPICE.

```
Multiple .END statements
*****
V1 1 0 SIN(0 1 1e3)
R1 1 2 1
R2 2 0 2
.TRAN 10u 1m
.PRINT TRAN V(1) V(2)
.END

V1 1 0 SIN(0 1 1e3)
R1 1 2 1
R2 2 0 3
.TRAN 10u 1m
.PRINT TRAN V(1) V(2)
.END
```

Both simulations will be run, once with the resistance of R2 equal to 2 and once with its resistance equal to 3. In Xyce, the simulation would only be run with the resistance of R2 equal to 2. All of the text after the first `.END` statement would be treated as comment lines by the Xyce parser. To run both simulations in Xyce, the appropriate `.STEP` or `.DATA` statement would be used to set the desired values for the resistance of R2.

5.2. Xyce `-hspice-ext` Command Line Option

The Xyce command line option, `-hspice-ext`, is almost always required when running a netlist translated by XDM. It allows the Xyce parser to accept a limited set of HSPICE syntax features, in lieu of the Xyce ones, for the limited set of cases shown in Table 5-1. These features are particularly difficult for XDM to translate, but are easy for Xyce to handle if it knows about them.

Table 5-1. `-hspice-ext` Command Line Option

Syntax	Functionality
<code>-hspice-ext units</code>	Toggles on $A=1e-18$ as a scaling factor
<code>-hspice-ext math</code>	Toggles in the HSPICE meanings for the logical operators <code> </code> and <code>&&</code> and the exponentiation synonym of <code>^</code> . It also uses the HSPICE functions for base10 and natural logarithms, which are <code>LOG10</code> and <code>LOG</code> .
<code>-hspice-ext random</code>	<code>AGAUSS()</code> and <code>GAUSS()</code> will return the mean value rather than a random variate
<code>-hspice-ext all</code>	Does all three

Table 5-1. -hspice-ext Command Line Option

Syntax	Functionality
<code>-hspice-ext units,math,random</code>	A comma-separated listing is also legal. This example is equivalent to all.

6. SPECTRE TRANSLATIONS: OVERVIEW AND KNOWN ISSUES

This is a list of common known issues the user may encounter when translating from Spectre. It is by no means an exhaustive list.

Spectre options All Spectre `options` lines will be commented out in the translated Xyce netlist for similar reasons as `.OPTIONS` in HSPICE (see preceding chapter "HSPICE Translations: Overview and Known Issues").

Analysis statements The only analysis statements XDM will currently translate are `dc`, `ac`, and `tran`. In addition, translations of secondary sweeps are not currently supported.

Instance parameters Guidelines similar to those listed for translations of HSPICE instance parameters (see preceding chapter "HSPICE Translations: Overview and Known Issues") should be kept in mind for translations of Spectre instance parameters as well.

Subcircuit parameters Spectre does not make a clear distinction between instance parameters of a `subckt` statement and parameters within that block. Because of this, the XDM translation puts all parameters inside the subcircuit block.

Spectre simulator command Spectre allows switching to a SPICE input reading mode through the use of the `simulator lang=spice` command. XDM can process this statement and will automatically switch to the HSPICE parser for translation in this case, although only one such language switch is allowed per file.

Multiplicity (M Factor) Similar issues surrounding multiplicity factor delineated in the previous chapter (see "HSPICE Translations: Overview and Known Issues") exists in Spectre translations as well.

Distribution functions and the statistics block Translations of functions that create random distributions in Spectre, such as `gauss` and `unif`, and the `statistics` blocks they reside in are not currently supported. They will be commented out in the translated files.

6.1. Commands known not to translate

The following commonly found statements in Spectre are not directly supported in Xyce:

- alter and altergroup
- checkpoint
- if

Verilog-A Support As mentioned in the previous chapter, Xyce does not generally support Verilog-A models. Therefore, XDM will comment out Spectre `.ahdl_include` statements.

7. TRANSLATION OF PDK MODEL LIBRARIES

The entirety of an HSPICE- or Spectre- based PDK model library can be converted by translating a circuit that includes the model library through the `.INC` or the `.LIB` file in HSPICE and the `include` file in Spectre, and using the `--auto` option.

To translate the PDK without an accompanying circuit, create a dummy file that has just single include statement for the top level wrapper file; for example in HSPICE:

```
.INC design.inc  
or  
.LIB design_wrapper.lib TT
```

and translate that with the `--auto` option.

It is recommended to translate a PDK's model library all at once with the circuit netlist using the `--auto` option, since XDM does some cross-checking of models and may remove parameters and/or comment out lines if all the information in a PDK isn't processed at the same time (see Chapter 5 for more details).

In past releases of Xyce, there were instances where it may have been necessary to pre-evaluate certain functions in the model library using XDM in order for a circuit to run in Xyce. If a Xyce simulation of a circuit hung and did not run to completion, users could try to translate the circuit with the pre-evaluation flag on. This is done using the `--eval` option in addition to the `--auto` option at the command line. As of release 7.2 of Xyce, this should no longer be necessary. However, the function pre-evaluation mode in XDM remains in available at this time.

- In this mode, XDM will evaluate functions in the model library based on the `.PARAM` statements and device instance parameters listed in the circuit, and will write the values of those evaluations into the circuit and model library files.
- The resulting translated model library will be specific to the circuit, and *cannot* be used in general with other circuits designed with the same model library. Users will have to re-run the translation again on different circuits to get the translated model library specific to that circuit.
- The function evaluation and pre-processing may take up to several minutes to complete, especially if the circuit includes functions that involve a high degree of nesting.

8. RELEASE NOTES

8.1. XDM 2.3

8.1.1. *General*

- Fixed bug where relative path names to files in .lib statements were incorrect.

8.1.2. *PSpice*

- XDM allows the undocumented PSpice syntax of having trailing commas in entries of “TABLE” statements.

8.1.3. *HSPICE*

- if/else/endif statements in HSPICE are now commented out since Xyce doesn’t currently support this.
- Node names containing brackets, as seen in HSPICE, are now allowed.
- Forward slashes in HSPICE device and subckt names are now allowed.

8.1.4. *Spectre*

- XDM now translates transient Spectre “sine” specifications.
- XDM comments out Spectre dc analysis statements with no parameters as it is not clear what the Xyce equivalent should be.
- XDM can now identify Spectre device instantiations of models defined in SPICE format.
- XDM removes Spectre “trise” parameter for R and C devices since this isn’t available in Xyce.
- Fixed bug in Spectre translation where sometimes the model name would not appear for a instantiation of a device of that model.
- Fixed XDM crash when processing some Spectre subcircuit definitions.
- Fixed bug in translation of Spectre “include” statements that have the parameter “section”. These will now be translated into “.LIB” statements in Xyce.”
- Added support for translation of Spectre “port” device instantiations. Not all instance parameters are translated at this time.
- Translate the “M” unit prefix (mega) in Spectre “AC” statements to the Xyce equivalent “X”.

- XDM comments out Spectre dc analysis statements with no parameters as it is not clear what the Xyce equivalent should be.
- Fixed bug where XDM aborted when translating Spectre “include” files with paths containing hyphens.

REFERENCES

- [1] Eric R. Keiter, Thomas V. Russo, Richard L. Schiek, Heidi K. Thornquist, Ting Mei, Jason C. Verley, Peter E. Sholander, and Karthik V. Aadithya. Xyce Parallel Electronic Simulator: Reference Guide, Version 7.2. Technical Report SAND2020-11841, Sandia National Laboratories, Albuquerque, NM, 2020.

APPENDIX A. Third Party Licenses

The Xyce XDM Netlist Translator makes use of code developed by various third parties. The following text is provided to comply with the licenses of the codes that require it.

XDM uses Boost:

Boost Software License - Version 1.0 - August 17th, 2003

Permission is hereby granted, free of charge, to any person or organization obtaining a copy of the software and accompanying documentation covered by this license (the "Software") to use, reproduce, display, distribute, execute, and transmit the Software, and to prepare derivative works of the Software, and to permit third-parties to whom the Software is furnished to do so, all subject to the following:

The copyright notices in the Software and this entire statement, including the above license grant, this restriction and the following disclaimer, must be included in all copies of the Software, in whole or in part, and all derivative works of the Software, unless such copies or derivative works are solely in the form of machine-executable object code generated by a source language processor.

XDM uses the Python 2.7 Standard Library:

Python is Copyright 2001–2020 Python Software Foundation; All Rights Reserved

1. This LICENSE AGREEMENT is between the Python Software Foundation ("PSF"), and the Individual or Organization ("Licensee") accessing and otherwise using Python 2.7.17 software in source or binary form and its associated documentation.
2. Subject to the terms and conditions of this License Agreement, PSF hereby grants Licensee a nonexclusive, royalty-free, world-wide license to reproduce, analyze, test, perform and/or display publicly, prepare derivative works, distribute, and otherwise use Python 2.7.17 alone or in any derivative version, provided, however, that PSF's License Agreement and PSF's notice of copyright, i.e., "Copyright © 2001–2020 Python Software Foundation; All Rights Reserved" are retained in Python 2.7.17 alone or in any derivative version prepared by Licensee.

3. In the event Licensee prepares a derivative work that is based on or incorporates Python 2.7.17 or any part thereof, and wants to make the derivative work available to others as provided herein, then Licensee hereby agrees to include in any such work a brief summary of the changes made to Python 2.7.17.
4. PSF is making Python 2.7.17 available to Licensee on an "AS IS" basis. PSF MAKES NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED. BY WAY OF EXAMPLE, BUT NOT LIMITATION, PSF MAKES NO AND DISCLAIMS ANY REPRESENTATION OR WARRANTY OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE OR THAT THE USE OF PYTHON 2.7.17 WILL NOT INFRINGE ANY THIRD PARTY RIGHTS.
5. PSF SHALL NOT BE LIABLE TO LICENSEE OR ANY OTHER USERS OF PYTHON 2.7.17 FOR ANY INCIDENTAL, SPECIAL, OR CONSEQUENTIAL DAMAGES OR LOSS AS A RESULT OF MODIFYING, DISTRIBUTING, OR OTHERWISE USING PYTHON 2.7.17, OR ANY DERIVATIVE THEREOF, EVEN IF ADVISED OF THE POSSIBILITY THEREOF.
6. This License Agreement will automatically terminate upon a material breach of its terms and conditions.
7. Nothing in this License Agreement shall be deemed to create any relationship of agency, partnership, or joint venture between PSF and Licensee. This License Agreement does not grant permission to use PSF trademarks or trade name in a trademark sense to endorse or promote products or services of Licensee, or any third party.
8. By copying, installing or otherwise using Python 2.7.17, Licensee agrees to be bound by the terms and conditions of this License Agreement.

XDM uses the RapidXML Library:

Copyright (c) 2006, 2007 Marcin Kalicinski

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

DISTRIBUTION

Hardcopy—Internal

Number of Copies	Name	Org.	Mailstop
1	Technical Library	9536	0899



Sandia
National
Laboratories

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.