Xyce™ Parallel Electronic Simulator
Version 7.2 Release Notes

Sandia National Laboratories

October 26, 2020

The **Xyce**™ Parallel Electronic Simulator has been written to support the simulation needs of Sandia National Laboratories' electrical designers. **Xyce**™ is a SPICE-compatible simulator with the ability to solve extremely large circuit problems on large-scale parallel computing platforms, but also includes support for most popular parallel and serial computers.

For up-to-date information not available at the time these notes were produced, please visit the **Xyce**™ web page at `http://xyce.sandia.gov`.

Contents

New Features and Enhancements

XDM

- place holder

New Devices and Device Model Improvements

- Support for BSIM-SOI 4.6.1 and 4.5.0 has been improved. Xyce now allows some configurations of the 5-, 6-, and 7- node variants of the model in addition to the 4-node provided by Xyce 7.1.
- The BSIM6 model now supports an optional fifth node, allowing netlist access to the temperature rise due to self-heating.
- Xyce now supports the DIODE_CMC model as the level 2002 diode.
- The lossless transmission line no longer forces a maximum timestep on the time integrator.

Enhanced Solver Stability, Performance and Features

- The expression library in Xyce has been completely rewritten and replaced. This new library is much more efficient for expressions found in modern process design kits (PDKs).

Interface Improvements

- .MEASURE is now supported for .NOISE analyses.
- The RISE, FALL and CROSS qualifiers are now supported for DERIV-WHEN, FIND-WHEN and WHEN measures for .AC, .DC and .NOISE analyses.
- The DNI and DNO operators can print out the individual input and output noise contributions for each noise source within a device.
- The -noise_names_file command line option can generate a listing of the noise source names for each device in a netlist.
- The P(*) and W(*) wildcards are now supported on .PRINT lines for TRAN and DC analysis modes.
- The I(*) wildcard is now supported on .PRINT lines for lead currents, for most Xyce devices, for TRAN and DC analysis modes. However, for multi-terminal devices, such as J, M, Q and Z, explict lead current designators (e.g, IC(*) for Q devices) must be used instead.
- The VR(*), VI(*), VP(*), VM(*), VDB(*), IR(*), II(*), IP(*), IM(*) and IDB(*) wildcards are now supported on .PRINT lines for branch currents for AC and NOISE analysis modes.
- There is now limited support for more complex wildcards, such as V(X1*), I(X1*), P(X1*) and W(X1*), on .PRINT lines.
- The new expression library (mentioned above) now fully supports complex numbers.
- The new expression library can now connect random operators such as AGAUSS to .SAMPLING analysis.

Xyce/ADMS Improvements

- Support for $port_connected has been added to enable designation of optional nodes.
- Support for $bound_step, enabling a device to impose a maximum step size on the simulation, has been added. This feature is primarily useful only for devices that act like independent sources.
- The deprecated, "legacy" Xyce/ADMS back-end that generated code that used Sacado for differentiation has been completely removed.
- The logic related to ddx() handling has been improved so that Xyce/ADMS is no longer emitting second derivative code when those derivatives are not needed. This results in dramatically improved compilation time of Verilog-A derived models.

Important Announcements

- The model interpolation technique described in the Xyce Reference Guide in section 2.1.18 has been marked as deprecated, and will be removed in a future release of Xyce.
- It has been determined that some distributions of Linux have broken builds of OpenMPI in their package repositories. Building Xyce from source code in parallel with these OpenMPI installs will result in a version of Xyce that may crash on some problems. This is not a bug in Xyce, but a packaging error of the OpenMPI package on those operating systems. Please see commentary in the "Known Defects" section of these release notes under bug number "967-SON".
- Xyce has deprecated the default conversion of quoted-string file names to a table of x,y pairs of data. The old convention of `PARAMETER="file.dat"` which worked in some model statements and in behavioral sources will now generate a warning in the Xyce output. The correct way to specify a file of data for a parameter is to use the new `tablefile` keyword as in `PARAMETER=tablefile("file.dat")`. While this release of Xyce will accept both the old and new syntax, the double quote technique will be removed in a future release, after which Xyce will only accept the syntax of `PARAMETER=tablefile("file.dat")`. Additionally, a new syntax of `PARAMETER=string("string value")` has been introduced to specify parameters that are pure strings. This will be deprecated in a future release and the simpler syntax of `PARAMETER="string value"` will be used to specify string valued parameters.
- The "Xygra" device, which was written as a special-purpose coupling mechanism to ALEGRA but which has occasionally been used for other coupling problems, has been removed. The new, more flexible "General External" device was created to take its place, and has supplanted the use of Xygra in ALEGRA. If your code has been using the Xygra capability to couple to Xyce, you must replace your usage with the new capability. The "General External" coupling mechanism is documented thoroughly in an application note available on the Xyce web site.

Interface Changes in this Release

Table 1: Changes to netlist specification since the last release.

Change	Detail
The precedence of FROM and TO versus AT was made identical for the DERIV and FIND measures.	The FROM and TO qualifiers now both take precedence over the AT qualifier for both DERIV and FIND measures. This behavior matches other simulators.
The RISE, FALL and CROSS qualifiers are no longer supported for TRAN-mode AVG, INTEG and RMS measures.	This changes makes the qualifier support for those three measure types consistent for all four measure modes (AC, DC, NOISE and TRAN). This is also consistent with HSPICE.
The Xyce-specific RFC_LEVEL qualifier is no longer supported for the DERIV-WHEN, FIND-WHEN and WHEN measures	That qualifier was added to improve the performance of the RISE, FALL and CROSS qualifiers for MAX, MIN and PP measures, and the performance of the Xyce-specific FRAC_MAX qualifier for TRIG and TARG measures. It was not particularly useful for WHEN measures.
The Points Value parameter on .AC and .NOISE lines must now be an integer greater than or equal to 1	Previously, the Point Value could be non-integer on .AC and .NOISE lines, but had to be non-negative. This change affects DEC and OCT sweeps, and prevents Xyce from picking a non-intuitive set of sweep values.
The Points parameter on .DC and .STEP lines must now be an integer	Previously, the Point parameter had no constraints for these two types of netlist commands. This change affects DEC and OCT sweeps, and prevents Xyce from picking a non-intuitive set of sweep values.

Defects Fixed in this Release

Table 2: Fixed Defects. Note that we have two different Bugzilla systems for Sandia users. SON, which is on the open network, and SRN, which is on the restricted network.

Defect	Description
Gitlab-ex 69: ksparse's alloc.c fails to compile with clang 12.0.0 on MacOS 10.15	The clang compiler supplied with Xcode 12 is more restrictive in complying with the C99 standard. As such, a bit of code in the ksparse solver package failed to compile. Fortunately, that bit of code was a relic of Spice 3f5, from which ksparse was copied. Removing the code allows Xyce to compile, without affecting the behavior of ksparse.
Gitlab-ex 56: Xyce header files unnecessarily include Trilinos header files	Xyce provides an API for coupling circuit simulation to external programs. The headers and shared libraries installed by "make install" should be sufficient to provide access to this feature, but some Xyce header files unnecessarily included Trilinos header files. This meant that users of Xyce shared libraries also needed to have access to complete sets of installed Trilinos headers. The header files needed to access the coupling API have been cleaned up and are now sufficiently self-contained to allow coupling of Xyce to an external code without the need for Trilinos headers or libraries. Trilinos is still required to build Xyce itself, and is still needed to build shared library plugins from Verilog-A models.
Gitlab-ex 54: Xyce/ADMS incorrectly sets "dependency" and "OPdependent" for analog function output variables	The internal ADMS data tree has an element called "dependency" that may be "constant", "noprobe", "linear", or "nonlinear". This element was being incorrectly set for analog function output variables (other than the return value). At the moment Xyce/ADMS does not use those elements for anything important, so the incorrect setting had no impact on generated C++ code. The incorrect setting could have had impact on future development of the Xyce/ADMS back-end.
Gitlab-ex 50: Xyce/ADMS mishandles potential contributions in presence of flow contributions	When Xyce/ADMS processed a module that contained both flow and potential contributions between the same two nodes, it was generating incorrect code that would not converge. This use case now emits code that puts the flow sources in parallel with the potential sources. This is not in compliance with the value retention rules of the Verilog-A language reference manual, which would have the second contribution of such a use case discard the information of the first contribution. The prior behavior, however, was not only non-compliant, but also broken. This issue impacted no standard Verilog-A models, as none make use of this type of contribution or rely on the value retention rules of the LRM.

Table 2: Fixed Defects. Note that we have two multiple issue tracking systems for Sandia Users. SON and SRN refer to our legacy open- and restricted-network Bugzilla system, and Gitlab refers to issues in our gitlab repositories.

Defect	Description
Gitlab-ex 42: Xyce/ADMS unnecessarily outputting second derivative code	An error in logic caused Xyce/ADMS to emit unnecessary code to compute second derivatives of variables. Second derivatives are only required when a variable whose value is computed via ddx() is subsequently used in non-noise contributions (a usage that is strongly discouraged and not present in any common models). This unnecessary code was often as large or larger than the required code, and resulted in extraordinary compilation times and compiler memory use.
Gitlab-ex 40: Xyce/ADMS incorrectly omitting variable declaration in obscure use cases	As a result of a design flaw that has been fixed in this release, previous versions of Xyce/ADMS would incorrectly omit the declaration of a module-scoped variable if the only use of that variable was in a lower-level block and an earlier block had a local variable of the same name.
Gitlab-ex 33: Lossless transmission line unnecessarily sets maximum time step	The Xyce lossless transmission line was implemented in a manner that restricted the time integrator's time step to never be longer than the transmission line's delay. This had been done to avoid applying extrapolation to the device history. This restriction severely impacted run time whenever a circuit included very short delay lines. This unnecessary restriction has been removed.
Gitlab-ex 24, 390-SON: Xyce/ADMS does not support $port_connected	The Verilog-A construct "$port_connected" tests whether an optional node has been specified on the instance line of a device. Using this construct anywhere in a Verilog-A model implicitly designates the named node as optional. Xyce/ADMS now supports this construct in a limited sense. See the Xyce/ADMS users guide for details.
812-SON: Incorrect response when N or V is used as a global param name	Use of a `.GLOBAL_PARAM` statement to define a parameter which had a name identical to a valid print line accessor function (e.g. "V" or "N") would result in error messages of the form "Function or variable ... is not defined" when that accessor was used on a print line, with the undefined variable being the node or device name inside the parentheses of the accessor. This was a bug in handling of global parameters, and the only workaround had been to avoid such parameter names. This workaround is no longer necessary.
1318-SON: Allow for numerical roundoff when FROM, TO or TD are expressions on .MEASURE lines	Xyce could give the incorrect answer when the `FROM`, `TO` or `TD` qualifiers on a .MEASURE line were expressions. This was caused by small numerical-roundoff errors in the calculation of the expression values.
1310-SON: Improve FROM and TO info printed to stdout for AC, DC and Noise measures	The information printed to stdout for the start and end points of the measurement window, for AC, DC and NOISE measures, is now correct for cases where the FROM and/or TO values are not equal to a sweep value. That information is also now correct for DERIV-WHEN, FIND-WHEN and WHEN measures.

Table 2: Fixed Defects. Note that we have two multiple issue tracking systems for Sandia Users. SON and SRN refer to our legacy open- and restricted-network Bugzilla system, and Gitlab refers to issues in our gitlab repositories.

Defect	Description
1307-SON: Improve Compatibility of .MEASURE with .DATA	If a `.DATA` statement is used to define a table-based sweep on a `.DC` line then `DC` measures now correctly use the row index in that table as their "swept variable". The `.MEASURE` section of the Xyce Reference Guide provides an example of this use case.

1306-SON: Fix FROM and TO qualifiers for DC mode measures	The `FROM` and `TO` qualifiers would work correctly if either both were used on a `.MEASURE DC` line, or if neither was used. Measures such as these now also work correctly for all supported DC measure types:
	`.measure dc maxFrom max v(1) FROM=4`
	`.measure dc minTo min v(1) TO=4`

1304-SON: Fix issues with FROM-TO qualifiers for DERIV-WHEN, FIND-WHEN and WHEN measures	These measure types could get the incorrect answer, for all measure modes, if the FROM or TO qualifiers were given. The interpolated WHEN time was not being properly compared with specified the FROM-TO window. Now, if the interpolated WHEN time is within the FROM-TO window then the measure value is valid. If the interpolated time is outside of the FROM-TO window then the measure is "FAILED".
1303-SON: Segfault in mixed signal interface	The Python method `getDeviceNames()`, and the underlying `xyce_getDeviceNames()` method in the mixed signal interface, would segfault if invoked for some model groups (e.g, D, L and M) when there were no devices from that model group in the Xyce netlist.
1176-SON: Expression tables have an efficiency bottleneck when tables are large	When a really large table (thousands of points) was used to specify a Bsrc via the expression library, there was a bottleneck which caused really slow execution.
1827-SRN: Expression library selection of minimum breakpoint distance is too large	The breakpointing algorithm for expression-based sources had a flaw in it that imposed an artificially large minimum distance between breakpoints. This prevented really fast rise times (for example) from working correctly.
794-SON: Bug in TABLE Form of Xyce Controlled Sources	In some case, a Xyce netlist that contains a controlled source that uses the TABLE form would get the correct answer at first. However, it may then "stall" (e.g, keep taking really small time-steps) and never complete the simulation run.
1222-SON: Near-infinite loop in expression library, uncovered by modern PDK files	Some modern process design kits (PDKs) contain very complex use of expressions, involving many nested function calls. In some cases, this was not parsable with the old Xyce expression library.

Known Defects and Workarounds

Table 3: Known Defects and Workarounds.

Defect	Description

Gitlab issue 85 Complex-valued parameters are not handed correctly	The Xyce expression library was rewritten for the 7.2 release, and has added support for complex numbers in expressions. However, the use of complex-valued parameters and global parameters is not correct yet. This is because parameters and global parameters are still assumed to always be real numbers. An example is:
	`.PARAM P1={log10(-2)}`
	`V1  1 0 1`
	`R1  1 0 1`
	`.OP`
	`.PRINT DC`
	`+ {Re(P1)} {Img(P1)}`
	`+ {Re(log10(-2))} {Img(log10(-2))}`
	`.END`
	The output will have `RE(P2)` equal 3.01e-01 and `IMG(P2)` equal 0, which is incorrect. However, the non-parameter fields will be output as `Re(log10(-2))` equal to 3.01e-01 and `Img(log10(-2))` equal to -1.36e+00, which is correct. The code assumes that the parameter `P1` is unconditionally real.

Gitlab issue 60 Xyce/ADMS omits derivative code for output arguments of analog functions if return value derivatives are not needed	Verilog-A permits analog functions (user defined functions) to have arguments that can be used to return values in addition to the return value of the function. These output arguments have their values calculated as a "side effect" of the function call. Due to a difficulty with bookkeeping, if the return value of the function is neither used in sources nor ddx() calls, Xyce/ADMS will not emit any code that calls the function in such a way that the derivatives of the output arguments would be computed. This can lead to incorrect results if the output arguments are later used in any way where their derivatives are required (e.g. on the right-hand side of non-noise contributions, or as the argument to be differentiated in a ddx() call). ***Workaround***: Either do not write analog functions with output arguments (thereby never having side-effects, a best practice), or make sure that the return value of the function is always used in a manner such that its derivatives will be required (use in non-noise contributions or as the argument to be differentiated by ddx().
Gitlab issue 28 Limitations on allowed parameter names is not fully documented	The exact limitations on allowed parameter names is not clear in the documentation, nor is any exhaustive list available. Single-character non-alphabetic names are mostly illegal for either `.param` or `.global_param` names, but there may be other undocumented limitations. These invalid parameter names will generally cause Xyce to exit with an appropriate error message.

Table 3: Known Defects and Workarounds.

Defect	Description

1309-SON: Incorrect results for AVG, INTEG, RMS measures when FROM and/or TO values are not equal to a time-step or sweep value	The `AVG`, `INTEG` and `RMS` measures can return an incorrect value if the `FROM` or `TO` qualifiers are given on the measure line and those values are not equal to an accepted time-step value, or one of the specified AC, DC or NOISE sweep values. A simple example for AC measures is:
	`.AC DEC 5 100Hz 1e6`
	`.MEASURE AC avg1  AVG VR(B) FROM=70e3`
	The answer will be correct if `FROM=`100e3, which is a requested AC sweep value. It will be incorrect for `FROM=`70e3.
	Workaround: A workaround is less obvious for TRAN measures. However, this `.OPTIONS` line can be used to force Xyce to take a time-step at the requested `FROM` and/or `TO` values:
	`.OPTIONS TIMEINT BREAKPOINTS=<fromValue>, <toValue>`

1262-SON: Duplicate L device definitions are not a parsing error when one of the duplicate L devices is part of a K device	As an example, this netlist will not produce a parsing error. Instead, the first L1 definition will be used in the K1 device definition.
	`* parsing fails to detect duplicate L1 devices`
	`V1 1 0 SIN(0 1 1KHz)`
	`L1 1 2 1e-3`
	`R1 2 0 1`
	`C1 2 0 1e-9`
	`* mutual inductor definition, with duplicate L1 device`
	`L1 4 0 1e-6`
	`L2 3 0 1mH`
	`K1 L1 L2 0.75`
	`.TRAN 0 1ms`
	`.PRINT TRAN V(1) v(2)`
	`.END`
	Workaround: There is none.

1241-SON: Expression library parsing bottleneck on large expressions	It has been determined that the expression library in Xyce can be the source of a severe parsing bottleneck when expressions are large and complex. Expressions of this sort show up most often when parsing large PDKs with complex use of the `.FUNC` feature, and when using the "tablefile" feature to import a large file of time/voltage pairs for use in a `B` source. ***Workaround***: There is currently no workaround for the issue of complex PDK function use, and the team is working on fixing this issue by redesigning the way Xyce handles expressions with user defined functions. For the "tablefile" issue, one should avoid using `B` sources with "tablefile" to read in large tables, and instead use the "PWL FILE" option of the `V` source, which does not have this parsing issue.

Table 3: Known Defects and Workarounds.

Defect	Description
1085-SON: Expression library mishandles `.FUNC` definitions of functions that begin with "I" and are two characters long	Xyce's expression library assumes that all terms of the form "`Ix(<arguments>)`" are lead current expressions, where "x" is either a lead designator such as "D", "G" , or "S" for a MOSFET or "C", "B", "E" for a BJT, or a digit indicating the pin number of the device associated with the lead. This assumption makes it impossible for users to define a function with a two-character name starts with "I". Unfortunately, the parser does not warn of this problem should a user define such a function, and the first indication of something being wrong is an unhelpful error message about an "undefined parameter or function" where the problematic function is used. ***Workaround***: Do not use function names of two character length that begin with the letter "I". If you are making use of a vendor-supplied library that includes definitions of functions such as "I0", you will have to modify the library to change the function name and all the instances of its use.
1037-SON: The use of non-constant values in .PARAM statements may lead to unexpected results	This netlist line (`.PARAM PA = {TEMP}`) is forbidden in Xyce since the special variable `TEMP` is not constant. However, that netlist line will not produce a Xyce parsing error, and the value of `PA` in the simulation may then be set to zero in some contexts. ***Workaround***: Non-constant values should only be used in `.GLOBAL_PARAM` statements in Xyce. This restriction may be different than in other Spice-like simulators.
1031-SON: .OP output is incomplete in parallel	When Xyce is run in parallel, the `.OP` output may be incomplete. ***Workaround***: One workaround is to run the netlist in serial. Another one is to use these command line options: `-per-processor -l output`. In that case, the per-processor log files will have the `.OP` information for the devices that were instantiated on each processor.
1009-SON: Transient adjoint sensitivities don't work with `.STEP`	Transient adjoint sensitivities require backward integrations to be performed after the primary transient forward integration. Doing this properly requires information to be stored during the forward solve, and for certain bookkeeping to be performed. Currently, these extra operations to support transient adjoints are not properly set up for `.STEP` analysis. ***Workaround***: None
1006-SON: SDT (expression library time integration) derivatives are not supported, so SDT can't be used for sensitivity analysis objective functions	SDT is a function supported by the Xyce expression library to compute numerical time integration. When this function is used, the expression library does not produce correct derivatives. This impacts Jacobian matrix entries, when SDT is used with a Bsrc, and it also impacts sensitivity analysis, when SDT is used in an objective function. For the former case, this can result in a lack of robustness for circuits that contain SDT-Bsrc devices. For the latter case, the objective function will simply be incorrect. ***Workaround***: None

Table 3: Known Defects and Workarounds.

Defect	Description

1004-SON: Ill-defined .STEP behavior for ”default parameters” for transient sources (SIN, EXP, PWL, PULSE and SFFM)	If, for example, these netlist lines are used in a transient (`.TRAN`) simulation:
	`V1 1 0 SIN(0 1 1)`
	`.STEP V1 1 2 1`
	then Xyce will run the simulation without warnings or errors, but no instance parameter of source `V1` will be stepped.
	Workaround: Explicitly use the desired stepped parameter (e.g., `V0`) on the `.STEP` line. For example, `.STEP V1:V0 1 2 1` would work correctly.

991-SON: Non-physical BH Loops in non-linear mutual inductor	Nonlinear mutual inductors that have high coupling coefficients (i.e. model parameter `ALPHA` over 1.0e-4) and low loss characteristics (i.e. zero `GAP`) can produce B-H loops with nonphysical hysteresis. ***Workaround***: Lower `ALPHA` values or larger `GAP` values can ameliorate this issue, but the root cause is still under investigation.
800-SON: Use of global parameters in expressions on .MEASURE lines will yield incorrect results	The use of global parameters in expressions on .MEASURE lines is not allowed, as documented in the Xyce Reference Guide. However, instead of producing a parsing error the measure statement will be evaluated with the specified qualifier value (e.g., `FROM`) being left at its default value. ***Workaround***: None, other than not doing this.
970-SON: Some devices do not work in frequency-domain analysis	Devices that may be expected to work in AC or HB analysis do not at this time. For AC this includes, but is not limited to, the lossy transmission line (LTRA) and lossless transmission line (TRA). For HB, the transmission lines do work but the nonlinear dependent sources (B source and nonlinear E, F, G, or H source) do not work when the expression is explicitly time-dependent. ***Workaround***: The LTRA and TRA models will need to be replaced with lumped transmission line models (YTRANSLINE) for AC analysis. There is not yet a workaround for the time-dependent B source in harmonic balance.

Table 3: Known Defects and Workarounds.

Defect	Description
967-SON: Zoltan segmentation fault with OpenMPI 2.1.x and 3.0.0 on some systems	It has been observed that when Xyce and Trilinos are built with OpenMPI 2.1.x or 3.0.0 on certain unsupported operating systems, a small number of test cases in the regression suite crash with a segmentation fault inside the Zoltan library. The Xyce team has determined that this is not a bug in either Xyce or Zoltan, but is instead due to some pre-packaged OpenMPI binaries on some operating systems having been built with an inappropriate option. This option, "–enable-heterogeneous" is explicitly documented in OpenMPI documentation as broken and unusable since 2013, but some package managers have OpenMPI binaries built with this option explicitly enabled. Turning on this option causes the resulting OpenMPI build to perform certain communication operations in a way that does not adhere to the MPI standard. There is nothing that can be done in Xyce or Zoltan to fix this issue — it is entirely a bug in the OpenMPI library as built on that system. A new test case has been added to the Xyce test suite in order to detect this problem. The test is "MPI_Test/bug_967", and it will be run whenever the test suite is invoked with the "+parallel" tag as described in the documentation for the test suite at `https://xyce.sandia.gov/documentation/RunningTheTests.html`. If this test fails, your system has a broken OpenMPI build that cannot be used with Xyce. At the time of this writing, this issue is present in Ubuntu Linux versions 17.10 and later, and there is an open bug report for it at `https://bugs.launchpad.net/ubuntu/+source/openmpi/+bug/1731938`. The issue may be present in other distros of Linux that are derived from Debian (as is Ubuntu), but we cannot confirm this. ***Workaround***: The only workaround for this problem is to build OpenMPI from source yourself, and not to include "–enable-heterogeneous" in its configure options. You should also post a bug report in your operating system's issue tracker requesting that they rebuild their OpenMPI binaries without the "–enable-heterogeneous" option. If you are using Ubuntu, you should register with that issue tracking system and add yourself to the list of people it affects in the existing bug report (doing so increases the "heat" of the bug, which may increase the likelihood of it being fixed).
964-SON: Compatibility of .PRINT TRANADJOINT with .STEP	The use of `.PRINT TRANADJOINT` is not compatible with `.STEP`. The resultant Xyce output will not be correct. ***Workaround***: There is none.

Table 3: Known Defects and Workarounds.

Defect	Description
932-SON: Analysis lines do not support expressions for their operating parameters	The Xyce parser and analysis handlers do not yet support the use of expressions on netlist analysis lines such as `.TRAN`. The parameters of these analysis lines (such as stop time for `.TRAN` or fundamental frequency for `.HB`) may only be expressed as literal numbers. ***Workaround***: There is no workaround internal to Xyce. Use of an external netlist preprocessor would be required.
883-SON .PREPROCESS REPLACEGROUND does not work on nodes referenced in expressions	The `.PREPROCESS REPLACEGROUND` feature does not replace ground synonyms if they appear in B source expressions. ***Workaround***: Do not use ground synonyms (`GND`, `GROUND`, etc.) in expressions. Use a literal "0" when referring to the ground node in expressions.
783-SON: Use of ddt in a B-Source definition may produce incorrect results	The `DDT()` function from the Xyce expression package, which implements a time derivative, may not function correctly in a B-Source definition. ***Workaround***: None.
727-SON: Xyce parallel builds hang randomly on OS X	During Sandia's internal nightly testing of the OSX parallel builds, we see that Xyce "hangs on exit" with an estimated frequency of less than 1-in-5000 simulation runs. We have not seen this issue with parallel builds for either RHEL6 or BSD. The hang is on exit, whether on a successful exit or on an error exit. The hang occurs after all of the Xyce output has occurred though. So, the user will get their sim results, but might have trouble if the individual Xyce runs are part of a larger script. ***Workaround***: None.
661-SON Lead currents and power accessors (I(), P() and W()) do not work properly in .RESULT Statements	There are two issues. First, `.RESULT` statements will fail netlist parsing if the requested lead current is omitted from the `.PRINT TRAN` line. As an example, this statement (`.RESULT I(R1)`) requires either `I(R1)`, `P(R1)` or `W(R1)` to be on the `.PRINT TRAN` line. Second, the output value, in the `.res` file, for the lead current or power calculation will always be zero.
583-SON: Switch with RON=0 leads to convergence failure.	The switch device does not prevent a user from specifying `RON=0` in its model, but then takes the inverse of this value to get the "on" conductance. The resulting invalid division will either lead to a division by zero error on platforms that throw such errors, or produce a conductance with "Not A Number" or "Infinity" as value. This will lead to a convergence failure. ***Workaround***: Do not specify an identically zero resistance for the switch's "on" value. A small value of resistance such as 1e-15 or smaller will generally work well as a substitute.
469-SON: Belos memory consumption on FreeBSD and excessive CPU on other platforms	Memory or thread bloat can result when using multithreaded dense linear algebra libraries, which are employed by Belos. If this situation is observed, either build Xyce with a serial dense linear algebra library or use environment variables to control the number of spawned threads in a multithreaded library.
468-SON: It should be legal to have two model cards with the same model name, but different model types.	SPICE3F5 and ngspice only require that model cards of the same type have unique model names. They accept model cards of different types with the same name. Xyce requires that all model card names be unique.

Table 3: Known Defects and Workarounds.

Defect	Description
250-SON: NODESET in Xyce is not equivalent to NODESET in SPICE	As currently implemented, `.NODESET` applies the initial conditions given throughout a full nonlinear solve for the operating point, then uses the result as an initial guess for a second nonlinear solve with no constraints. This is not the same as SPICE, which merely applies the given initial conditions to a single nonlinear solve for the first two iterations, then lets the problem converge with no further constraints. This can lead to a Xyce `.NODESET` failing where the same netlist in SPICE might not, if the initial conditions are such that a full nonlinear solve cannot converge with those constraints in place. There is no workaround.
247-SON: Expressions don't work on .options lines	Expressions enclosed in braces ({ }) are handled specially throughout Xyce, and may only be used in certain contexts such as in device model or instance parameters or on `.PRINT` lines.
49-SON Xyce BSIM models recognize the model TNOM, but not the instance TNOM	Some simulators allow the model parameter TNOM of BSIM devices to be specified on the instance line, overriding the model parameter TNOM. Xyce does not support this.
27-SON: Fix handling of .options parameters	When specifying .options for a particular package, what gets applied as the non-specified default options might change.

2119-SRN: Voltages from interface nodes for subcircuits do not work in expressions used in device instance parameters	This bug can be illustrated with this netlist fragment:
	`X1 1 2 MySub`
	`.SUBCKT MYSUB a c`
	`R1   a b 0.5`
	`R2   b c 0.5`
	`.ENDS`
	`B1 3 0 V={V(X1:a)}`
	This fragment will produce the netlist parsing error `Directory node not found: X1:A`. The workaround is to use `V={V(1)}` in the B-source expression instead. This bug also affects the solution-dependent capacitor.

1923-SRN: LC lines run out of memory, even if equivalent (larger) RLC lines do not.	In some cases, circuits that run fine using an RLC approximation for a transmission line, exit with an out-of-memory error if the (supposedly smaller) LC approximation is used.
1595-SRN: Xyce won't allow access to inductors within subcircuits for mutual inductors external to subcircuits	It is not possible to have a mutual inductor outside of a subcircuit couple to inductors in a subcircuit. ***Workaround***: Put all inductors and mutual inductance lines that couple to them together at the same level of circuit hierarchy.

Supported Platforms

Certified Support

The following platforms have been subject to certification testing for the Xyce version 7.2 release.

- Red Hat Enterprise Linux® 7, x86-64 (serial and parallel)
- Microsoft Windows 10®, x86-64 (serial)
- Apple® macOS 10.14 and 10.15, x86-64 (serial and parallel)

Build Support

Though not certified platforms, Xyce has been known to run on the following systems.

- FreeBSD 11.x on Intel x86-64 and AMD64 architectures (serial and parallel)
- Distributions of Linux other than Red Hat Enterprise Linux 6
- Microsoft Windows under Cygwin and MinGW.

Xyce Release 7.2 Documentation

The following Xyce documentation is available on the Xyce website in pdf form.

- Xyce Version 7.2 Release Notes (this document)
- Xyce Users' Guide, Version 7.2
- Xyce Reference Guide, Version 7.2
- Xyce Mathematical Formulation
- Power Grid Modeling with Xyce
- Application Note: Coupled Simulation with the Xyce General External Interface
- Application Note: Mixed Signal Simulation with Xyce 7.2

Also included at the Xyce website as web pages are the following.

- Frequently Asked Questions
- Building Guide (instructions for building Xyce from the source code)
- Running the Xyce Regression Test Suite
- Xyce/ADMS Users' Guide
- Tutorial: Adding a new compact model to Xyce

External User Resources

- Website: `http://xyce.sandia.gov`
- Google Groups discussion forum: `https://groups.google.com/forum/#!forum/xyce-users`
- Email support: xyce@sandia.gov
- Address:

  Electrical Models and Simulation Dept.
  Sandia National Laboratories
  P.O. Box 5800, M.S. 1177
  Albuquerque, NM 87185-1177

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and 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.
SAND2020-11776 O