TPT Severe Issues

Introduction

The following document contains a list of known severe issues of TPT. By severe issues we mean issues/bugs in particular versions of TPT that:

1. might cause malfunctions in the behavior of TPT
2. are hard or even impossible to find by the TPT user herself/himself
3. cause the risk that bugs/defects in a SUT (system under test) are not detected by TPT in cases where TPT would have been able to reveal these bugs/defects in the SUT without the aforementioned malfunction in the behavior of TPT.

Usually these severe issues address the situations where the problem might appear and have well-defined workarounds.

ISSUE # 30102

TITLE:
When comparing INT64 signals using Min/Max or Signal Comparison assesslet with values larger than 2^53 or smaller than -2^53, the computation can incorrectly compare the signal with the reference(s) which might lead to PASSED results even if the specified bounds are exceeded.

ISSUE DETECTION:
11-January-2021

AFFECTED VERSIONS OF TPT:
TPT 8 - TPT 16

PRECONDITIONS:
The Min/Max or Signal Comparison assesslet is used with INT64 signals with values larger than 2^53 or smaller than -2^53.

DETAILS:
If signal or reference values are larger than 2^53 or smaller than -2^53, the difference can be missed because the values are converted and compared as double values. (Since values larger/smaller than 2^53/-2^53) cannot be converted to double without losing precision, floating point precision problems might occur in such cases.)

EFFECT OF THE ISSUE:
Values outside of the specified bounds might be overlooked by TPT leading to PASSED results, but should be FAILED.

WORKAROUND:
Avoid using INT64 signals in Min/Max or Signal Comparison assesslets if the values are larger than 2^53 or smaller than -2^53.

RESOLVED IN:
TPT 15u4, TPT 16u1

ISSUE # 30063

TITLE:
When iterating in assessment scripts via an inlined loop and

applying signal processing functions like TPT.average(), TPT.min(),
... with changing argument in each loop, the result of the first iteration
is incorrectly used for the following iterations.

ISSUE DETECTION:
16-December-2020

AFFECTED VERSIONS OF TPT:
TPT 8 - TPT 16

PRECONDITIONS:
The usage of the signal processing function must be applied on an expression
or a signal that is being changed while iterating through an inlined loop or list
comprehension.

DETAILS:
For faster computation of timed expressions of the form
  foo(t) := TPT.average(...)
results of signal processing functions are being cached.
The cached result is invalidated as soon as a new line is reached.
When iterating through an inlined loop, the same expression is evaluated multiple
times.
If during this inlined iteration a variable of the expression is changed,
the cached result of the signal processing function is used instead of a newly
calculated.

Affected are inlined expressions of the form
  for x in range(n): print TPT.min(...+x)
  while x < n: x=x+1;print TPT.min(...+x)
as well as list comprehension:
  my_list = [TPT.min(...+x) for x in range(n)].

EFFECT OF THE ISSUE:
Old cached values are used instead of recalculating the value in every iteration,
which results in wrong computation results.

WORKAROUND:
Avoid using inlined loops or list comperhensions and use loops with indentation
in multiple code lines instead.

RESOLVED IN:
TPT 15u4 TPT 16u1

ISSUE # 26884
==========
TITLE:
Local search and replace in step lists always replaced all
occurrences in table steps at once and so may replace entries
unnoticeably the user did not intend to change.

ISSUE DETECTION:
15-Jul-19

AFFECTED VERSIONS OF TPT:
TPT 12 – TPT 13

PRECONDITIONS:
A step list including a table step with multiple occurrences
of a search term and trying to replace one of these.

DETAILS:
If a search term occurs multiple times in a table step trying to replace one of them using the search dialog always replaces all occurrences in the table step. Generally, the issue affects all tables embedded in other tables but the table step is the only one where this condition applies.

EFFECT OF THE ISSUE:
Due the unexpected behavior of search and replace more changes than intended may be done to table steps leading to unintended TPT model behavior.

WORKAROUND:
Using global search and replace to replace occurrences of a search term in table steps. Changes done by local search and replace affecting table steps have to be reviewed carefully.

RESOLVED IN:
TPT13u2

ISSUE # 26796
==========
TITLE:
If explicit array-of-struct values are being used in TPT models (e.g. “mychannel := [{2,3},{4,param+5}]”) these expressions can compute wrong values at runtime if the arrayof-struct expression refers to non-constant channels or parameters.

ISSUE DETECTION:
01-Jul-19

AFFECTED VERSIONS OF TPT:
TPT 12 — TPT 13

PRECONDITIONS:
TPT computes wrong values at runtime under the following conditions: an explicit array-of-struct value is assigned to a channel or parameter (e.g. in a step list) AND the array-ofstruct value itself depends on channels/parameters AND these channels/parameters are not constant (at runtime) AND these channels/parameters have values that differ from their default values (as specified in the declaration editor).

DETAILS:
In the special case of explicit array-of-struct expressions (e.g. “mychannel := [{2,3},{4,param+5}]”) any reference to channels/parameters inside this expression will not be computed dynamically (at runtime), but computed just once when initializing the expression (before runtime). Therefore, if the value of channels/parameters that are referred inside this expression changes at runtime, the array-of-struct expression will not update its value. The fix now explicitly prohibits the usage of (a) references to all channels (b) references to all parameters that are modified dynamically in the test model inside any explicit array-of-struct expression. Otherwise the TPT compiler will raise a compile error.

EFFECT OF THE ISSUE:
In the special case of explicit array-of-struct expressions
(e.g. “mychannel := [{2,3},{4,param+5}]”) with references to
non-constant channels or parameters, the value of the arrayof-struct expression
gets stuck with its initial value for
all points in time even if the referred channels/parameters
change. After the fix related to this issue, those expressions
are denied by the compiler.

WORKAROUND:
Instead of assigning the whole array-of-struct use individual
assignments for non-constant elements.

RESOLVED IN:
TPT13u2

ISSUE # 24165
==========
TITLE:
TPT-VMAPI corrupts values at runtime for array input or read/
write output channels that have a specified dimension in TPT
less than the specified dimension in the SUT(C code).

ISSUE DETECTION:
11-Jun-18

AFFECTED VERSIONS OF TPT:
TPT 8 — TPT 12

PRECONDITIONS:
Test execution by means of EXE-platform, CANoe-platform,
ASCET-platform (excluding ASCET@FUSION), C-code-platform.
Test model contains array channels (including struct
channels containing arrays or array channels containing
structs) that are configured as input channels. The dimension
of these array input channels in TPT is smaller than the
dimension in the SUT.

DETAILS:
Assume that the test model uses the feature of TPT which
allows the dimension of array channels in TPT (logical dim)
to be smaller than the dimension in the SUT (physical dim). If
one or many of such the array channels are read from the SUT
to TPT per test cycles (periodically) during test execution
TPT potentially corrupts some *other* channels/parameter
values at runtime. For each test model the number of affected
channels/parameters is fixed, but unpredictable.

EFFECT OF THE ISSUE:
Channels and/or parameters can be corrupted at runtime under
the specified preconditions. The signals recorded for these
channels/parameters in the TPTBIN file are corrupted
afterwards and the subsequent assessment will analyze the
results based on these potentially corrupted data.

WORKAROUND:
Regenerate the testframe.c testdriver to ensure that the
dimension of signals in TPT matches the dimension of signals
in the SUT.

RESOLVED IN:
TPT11u3, TPT12u2

ISSUE # 29261
==========
TITLE:
When executing tests using the FUSION platform,the mechanism "Read parameters from FUSION only once (before first test case)" does not ensure that the first test case has completed the parameter exchange before the following test cases use the parameters.

ISSUE DETECTION:
4—September-2020
AFFECTED VERSIONS OF TPT:
TPT 12, TPT 13, TPT 14, and TPT 15

PRECONDITIONS:
The checkbox "Read parameters from FUSION only once (before first test case)" must be selected in the FUSION platform and the number of cores in the Execution Configuration dialog must be greater than 1.

DETAILS:
When running tests in multicore mode using the FUSION platform and the option "Read parameters from FUSION only once (before first test case)" is selected, the test cases are executed immediately without waiting until the parameters have been exchanged.

EFFECT OF THE ISSUE:
For tests executed in parallel (i.e. all but the first test case), not the parameter values determined during a parameter exchange are used, but the default values from the Declaration Editor (nondeterministic behavior).

WORKAROUND:
Deselect the "Read parameters from FUSION only once (before first test case)" checkbox in the FUSION platform configuration, or do not run tests in multicore mode.

RESOLVED IN:
TPT14u3, TPT15u2