OCC Flow for simulation

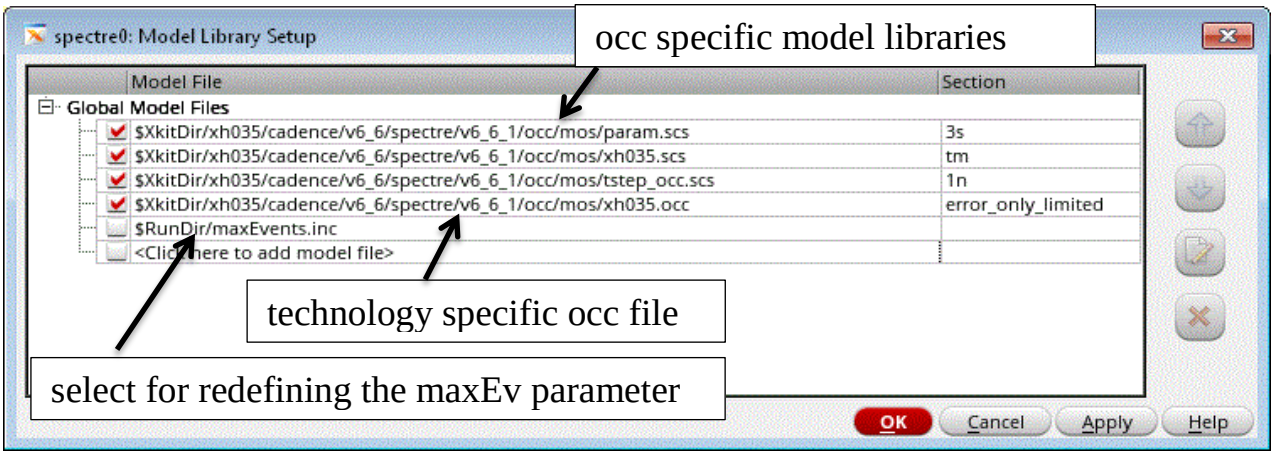
How to perform an OCC (Operating Condition Check) – simulation

The setup steps for the supported simulators (spectre, hspice and eldo) are nearly the same. The steps to open the model library setup are depending on the environment and the simulation tool. For more details please refer to the manuals of the used tool.

The setup is described on the example of Spectre

- Invoke “Cadence Analog Design Environment

- Choosing “Simulator/Directory-..” -> Spectre as preferred simulator
- Choosing “Setup” -> Model Libraries …”
- Change the normal library paths to the OCC-paths
  Add the both Occ special libraries to the model paths.
  If the standard model path is used only the “occ” subpath must be included in the model paths and the two model libraries “tstep_occ.lib” and <tech>.occ (in the example below “xh035.occ”) must be added to the model libraries.



The first two lines show the normal model libraries with the “occ” extensions. In the tstep_occ library a simulation variable is defined which influence the display of the occ events. The library <tech>.occ initialized the VerilogA environment.

For this library following sections are possible:

error_unlimited	only the min/max values are checked
multi_range_unlimited	additional the ranges defined by absmin and absmax (see PDS of the technology) are checked

If the VerilogA files contains an error limitation (only the first n violations of every checked parameter for each instance are reported) additional two sections are available:

error_limited	Only the first n changes are reported n is preset but configurable by the user
multi_range_limited	

The max reported errors n is defineable with the parameter maxEv. This parameter can be redefined manually or by using the file maxEvents.inc (see last line in the picture).
A template for this file is located in the model directory. For using this file make a copy in your environment ($RundDir) and change the the value of the parameter.

In most cases an occ model set for checking multiple ranges is defined. That means a SOA-range is defines, a lightly dangerous area (the device can work in these areas with special conditions which are defined in the Process Specification of the technology) and a forbitten error region is defined. To select this “multi range” definitions use the section ‘multi_range_(un)limited’.

- Start transient simulation within the “Cadence Analog Design Environment” windows

- For a simulation with Spectre without the “Cadence Analog Design Environment” the netlist must be prepared.

```
simulator lang=spectre
global 0

include "<path>/occ/mos/xh035.scs" section=tm
include "<path>/occ/mos/tstep_occ.scs" section=s1n
include "<path>/occ/mos/xh035_mr.occ"

…
…
…
tran tran stop=10u write="spectre.ic" writefinal="spectre.fc" \
…
…
saveOptions options save=allpub
```

- evaluate simulation results of the created “occ.err” log file which is located in the simulation project directory e.g. “/simulation/…/occ.err”

- The “occ.err” text file (see example below) contains all violations regarding voltage limitations (“OCA”) of each primitive device with the associated time step related to operating conditions define within the process specification.

Device	OC-Para	Event	Limit	Time	Instance
Cpoly	Vpm	leaving	OCA (> 5.5 V)	at 0.000000e+00	I3.C0.occ_m1
pmos4	VDpsub	leaving	OCA (> 5.5 V)	at 0.000000e+00	I2.M18.occ_m1
pghv	VDB	leaving	OCA (> 0.5 V)	at 1.590018e-06	I3.M3.occ_m1
cpoly	Vpm	entering	SOCA (-5.5 V .. 5.5 V)	at 3.679900e-06	I3.C0. occ_m1
nhv	VSB	leaving	OCA (> 11.0 V)	at 4.654617e-06	I3.M1.occ_m1
nhv	VSB	leaving	OCA (> 11.0 V)	at 4.654617e-06	I3.M2.occ_m1
pghv	VDB	entering	SOCA (-25.0 V .. 0.5 V)	at 6.545138e-06	I3.M3.occ_m1
nhv	VSB	entering	SOCA (-0.5 V .. 11.0 V)	at 2.006150e-03	I3.M1.occ_m1
nhv	VSB	entering	SOCA (-0.5 V .. 11.0 V)	at 2.006150e-03	I3.M2.occ_m1
cpoly	Vpm	leaving	OCA (> 5.5 V)	at 2.009185e-03	I3.C0.occ_m1
pghv	VDB	leaving	OCA (> 0.5 V)	at 5.001485e-03	I3.M3.occ_m1
cpoly	Vpm	entering	SOCA (-5.5 V .. 5.5 V)	at 5.003387e-03	I3.C0.occ_m1
nhv	VSB	leaving	OCA (> 11.0 V)	at 5.004651e-03	I3.M1.occ_m1
nhv	VSB	leaving	OCA (> 11.0 V)	at 5.004651e-03	I3.M2.occ_m1
pghv	VDB	entering	SOCA (-25.0 V .. 0.5 V)	at 5.006209e-03	I3.M3.occ_m1

- The “occ.err” file indicates “**leaving**” and “**entering**” of the OCA related to the operating conditions of all terminal voltages defined in the Process Specification.
- Entering SOCA means the device works in the save area
- Entering OCAn n=1,-1… means the device works in a dangerous area.
- Leaving OCA means the device works in a forbidden area and in a real environment it would be destroyed.

- the circuit designer has the complete **“responsibility” to evaluate and to interpret** the existing events / violations

- The OCV-Tools helps to visualize the error distribution and shows the results per device in a table.

- the “OCC” simulation method is only a kind of terminal voltage monitor of primitive devices within a circuitry

- experiences show, problem devices are mostly not devices which are directly connected with the power supply rails but primitive devices of intermediate circuit nodes are more jeopardized (e.g. compensation capacitors or capacitive coupling)

Eldo and Hspice direct netlist preparing

Change the model path to the occ models. The rules how to change to the occ models are the same as described above in the Spectre section. Next add the library tstep_occ.lib with the .lib command and <tech>.occ file with the .inc command to the net list (see example).
In the examples <sim> stands for “eldo” or “hSpiceS”
Example for Variant 1
Change model path from
.lib “…/<sim>/xh035/mos/bsim3v3.lib” tm
…
to
.lib "…/<sim>/xh035/occ/mos/bsim3v3.lib" tm
…
Add new statements
.lib "…/<sim>/xh035/occ/mos/tstep_occ.lib" 10n
.inc "…/<sim>/xh035/occ/mos/xh035.occ"

The simulation run creates the log file “occ.err” in the run directory. The content of this file is the same as described in the spectre section above. The format is lightly different depend on the used simulator.

Additional to the VerilogA based Occ check setup files for the simulator build in parameter checks are available.

The files names are <tech>.<sim>Soa
tech = technology e.g. xt018, xh035 ..
sim = used simulator (spectre, eldo, hSpice)

Please use the description in the simulator manual to setup and enable these checks.

For Spectre use the description of Assert.
For Eldo use the description of SOA check
For Hspice use the description of .biaschk.