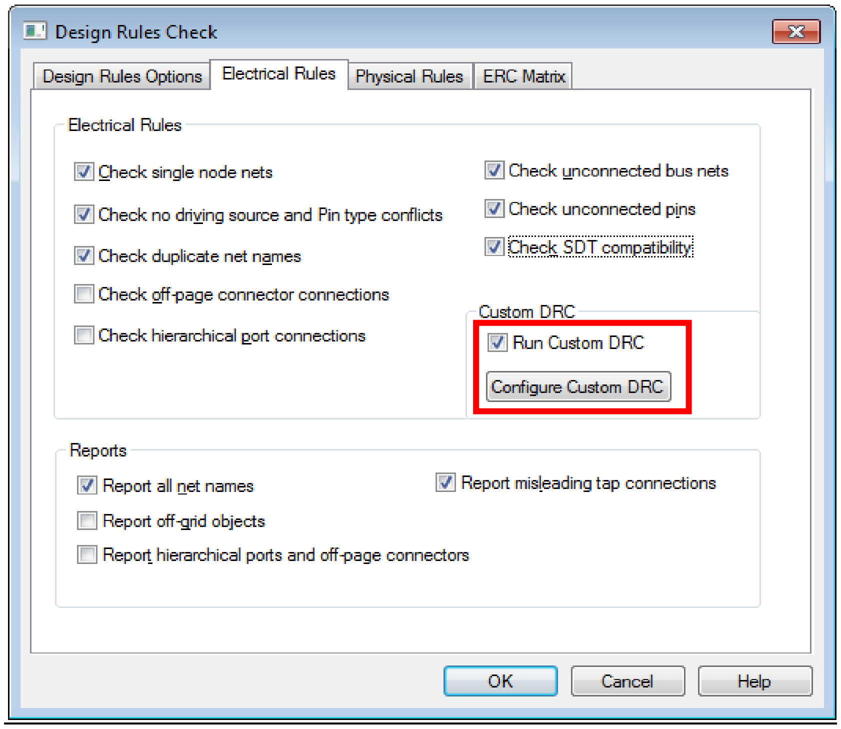
<u>Custom DRC</u>

"Custom DRC" functionality is a TCL enabled DRC which allows user to add their own DRC's.



Design Rules Check

Design Rules Options | Electrical Rules | Physical Rules | ERC Matrix

Electrical Rules

- [✓] Check single node nets
- [✓] Check no driving source and Pin type conflicts
- [✓] Check duplicate net names
- [ ] Check off-page connector connections
- [ ] Check hierarchical port connections
- [✓] Check unconnected bus nets
- [✓] Check unconnected pins
- [✓] Check SDT compatibility

Custom DRC

- [✓] Run Custom DRC

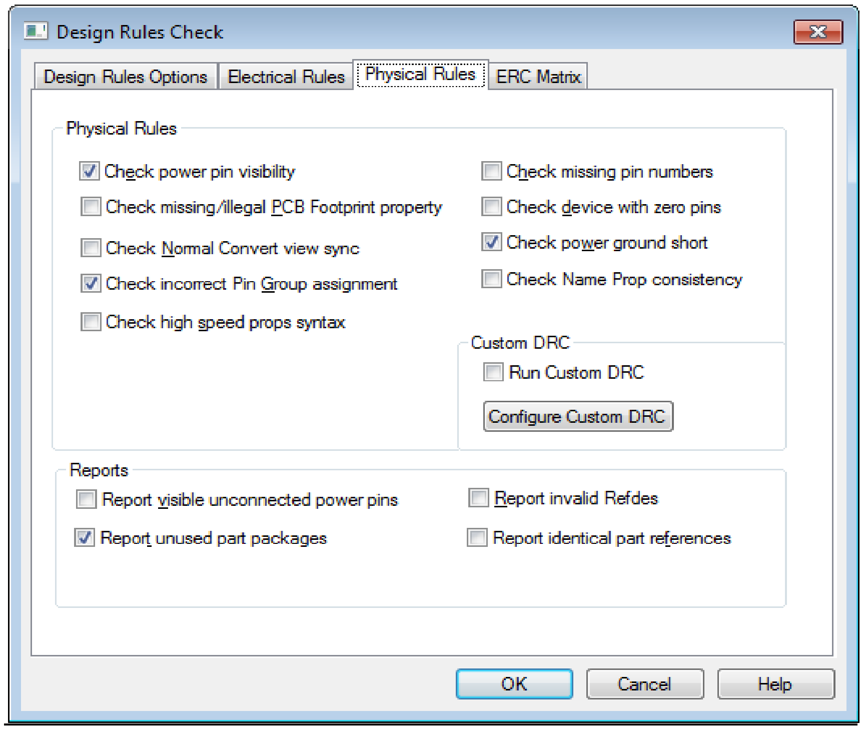
Configure Custom DRC

Reports

- [✓] Report all net names
- [ ] Report off-grid objects
- [ ] Report hierarchical ports and off-page connectors
- [✓] Report misleading tap connections

OK | Cancel | Help

<u>User can click "Run Custom DRC" checkbox to enable Custom DRC's. User can also configure the DRC's to be executed through "Configure Custom DRC" Button.</u>



Design Rules Check

Design Rules Options | Electrical Rules | Physical Rules | ERC Matrix

Physical Rules

- [✓] Check power pin visibility
- [ ] Check missing/illegal PCB Footprint property
- [ ] Check Normal Convert view sync
- [✓] Check incorrect Pin Group assignment
- [ ] Check high speed props syntax
- [ ] Check missing pin numbers
- [ ] Check device with zero pins
- [✓] Check power ground short
- [ ] Check Name Prop consistency

Custom DRC

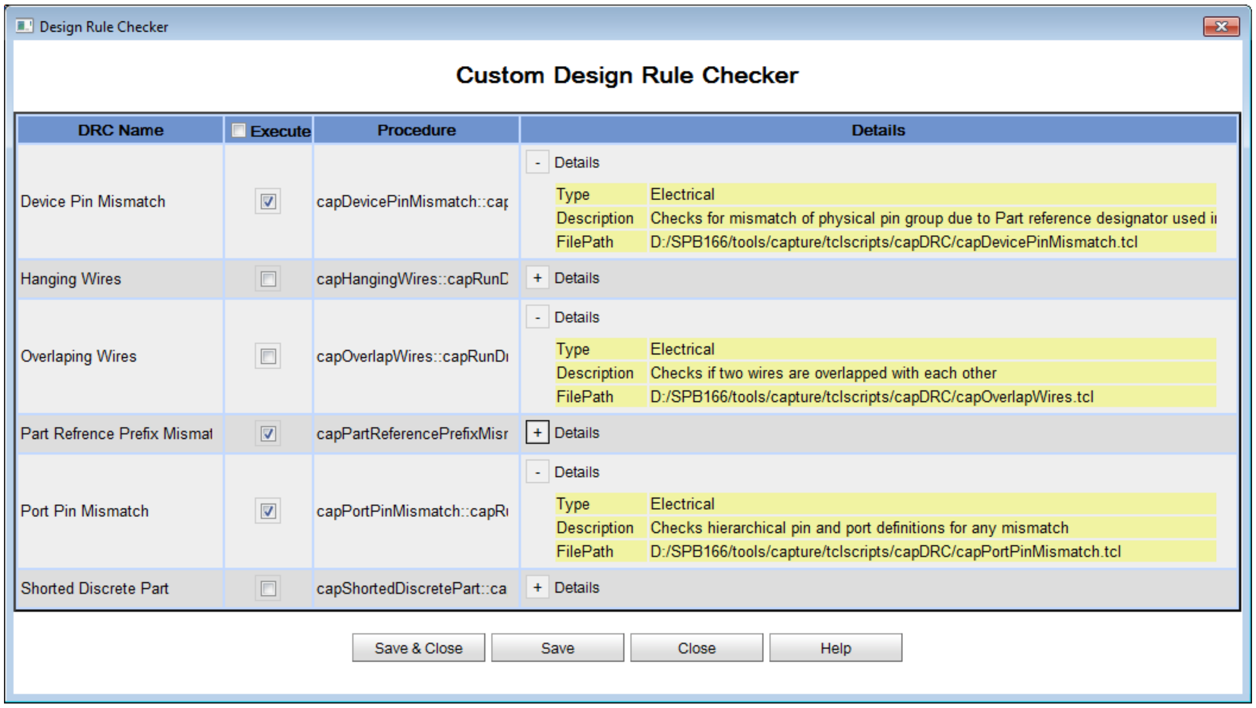
- [ ] Run Custom DRC

Configure Custom DRC

Reports

- [ ] Report visible unconnected power pins
- [✓] Report unused part packages
- [ ] Report invalid Refdes
- [ ] Report identical part references

OK | Cancel | Help



Design Rule Checker

Custom Design Rule Checker

DRC Name	☐ Execute	Procedure	Details
Device Pin Mismatch	✓	capDevicePinMismatch::cap	- Details Type: Electrical Description: Checks for mismatch of physical pin group due to Part reference designator used i FilePath: D:/SPB166/tools/capture/tclscripts/capDRC/capDevicePinMismatch.tcl
Hanging Wires	☐	capHangingWires::capRunD	+ Details
Overlaping Wires	☐	capOverlapWires::capRunD	- Details Type: Electrical Description: Checks if two wires are overlapped with each other FilePath: D:/SPB166/tools/capture/tclscripts/capDRC/capOverlapWires.tcl
Part Refrence Prefix Mismat	✓	capPartReferencePrefixMisr	+ Details
Port Pin Mismatch	✓	capPortPinMismatch::capR	- Details Type: Electrical Description: Checks hierarchical pin and port definitions for any mismatch FilePath: D:/SPB166/tools/capture/tclscripts/capDRC/capPortPinMismatch.tcl
Shorted Discrete Part	☐	capShortedDiscretePart::ca	+ Details

Save & Close · Save · Close · Help

<u>User Selects the DRC's to be executed. Once Configured Configurations are saved in INI file which are honored every time user runs Custom DRC .</u>

<u>Steps to Add new DRC</u>

Create a TCL File and corresponding entry in pkgIndex.tcl in any folder under tclscripts.

Add the Following Methods in the tcl file with any desired namespace and modify according to the DRC.

```
proc ::<NAMESPACE>::<METHOD1>{ args } {
        set lScope [lindex $args 0 0]
        set lMode [lindex $args 0 1]
        set lCreateDrcMarkers [lindex $args 0 2]
        set lLogFilePath [lindex $args 0 3]
        capCustomDRC::capSetCreateMarker $lCreateDrcMarkers

        set lMessage "\ Running <NAMESPACE>::<METHOD1> \n\n"
        # Setting the Variables for logging
        capCustomDRC::capSetLogFilePath $lLogFilePath
        capCustomDRC::capCustomDrcLog $lMessage

        capProcessDRC::capProcessSelection "<NAMESPACE>" $lScope $lMode
}
```

```
proc ::<NAMESPACE>::<METHOD2 >{} {
	return true
}

proc ::<NAMESPACE>::<METHOD3>{} {
	package require orPrmWebComp
	set lDrcName "<DRC NAME as any text string>"
	set lProc " <NAMESPACE>::<METHOD1>"
	set lIsExecute [capCustomDRC::capCustomElectricalDrcFindExecutableStatus $lProc]
	set lOptional [DboTclHelper_sMakeStdVector]
	DboTclHelper_sPushBackToVector $lOptional "Type"
	DboTclHelper_sPushBackToVector $lOptional "Electrical" #valid values = Electrical/Physical
	DboTclHelper_sPushBackToVector $lOptional "Description"
	DboTclHelper_sPushBackToVector $lOptional "<Any Description>"
	DboTclHelper_sPushBackToVector $lOptional "FilePath"
	set lFilePath [file join $::capShortedDiscretePart::scriptDir <TCL_FILE_NAME>]
	DboTclHelper_sPushBackToVector $lOptional $lFilePath
	set lReturn [CapCustomDRCElectricalAddItem $lDrcName $lIsExecute $lProc $lOptional]
}
```

Add below line in tcl file and modify according to DRC:

```
RegisterAction "_cdnCapCustomDRCElectricalAddItem" "::<NAMESPACE>::<METHOD2>" ""
"::<NAMESPACE>::<METHOD3>" ""
# RegisterAction "_cdnCapCustomDRCPhysicalAddItem" "::<NAMESPACE>::<METHOD2>" ""
"::<NAMESPACE>::<METHOD3>" "" #for Physical DRC's
```

Now Define functions according to your requirements which are called from "capProcessDRC.tcl". These functions are called in catch statements, so rest undefined functions will be ignored. For Ex:

```
#proc ::<NAMESPACE>::capProcess<ObjectType> { pObject } { # e.g.

proc ::capHangingWires::capProcessWire { pWire } {
	set lsearchIndex [lsearch $::capHangingWires::WireList $pWire]
	if { $lsearchIndex == -1 } {
		capHangingWires::capProcessWireObtained $pWire
		lappend ::capHangingWires::WireList $pWire
	}
}
```

Add an init file in capAutoLoad folder with the following content.

```
if { [catch {package require <NAMESPACE>}] } {
} else {
}
```