

ModelSim PE and the Swift Interface

1 USING THE SWIFT INTERFACE WITH MODEL SIM PE

If you are a VHDL customer and you need to simulate the RocketIO (gt), Ethernet MAC (emac) or PowerPC (PPC405) cores for Xilinx then you will need to use the Swift Interface feature. These cores are supplied as SmartModels, i.e. they are written in C, and neither the Xilinx OEM edition nor the basic configuration of ModelSim PE support the Foreign Language Interface (FLI) which is required to simulate C code with VHDL. To allow ModelSim PE (the Swift Interface is not supported in XE) to simulate the core you will need a Swift Interface feature (msimpeswift). Verilog customers do not need the extra feature because the PLI (Programming Language Interface) is specified as part of the Verilog standard and therefore automatically included in PE.

2 SETTING UP THE SWIFT INTERFACE

To set up and check the Swift Interface you will need to do the following:

1. Add the Swift Interface feature (msimpeswift) to your licence if not already there.
2. Set the LMC_HOME environment variable:
`LMC_HOME = %XILINX%\smartmodel\nt\installed_nt\`
3. Set or edit the PATH environment variable to include:
`PATH = <ModelSim_Install>\win32pe;%LMC_HOME%\bin;%LMC_HOME%\lib\pcnt.lib`
4. Compile libraries. For Xilinx you can use the `complib` batch command (`complib -help` will give you all the options) from a DOS command prompt e.g.:
`complib -s mti_pe -f all -l vhdl -w -o <compile_location>`
Note: If more than one instance of ModelSim referenced on the PATH variable you will also need the `-p` switch.
5. Edit the [lmc] section of the modelsim.ini file (in the working directory) as shown in section 3.
6. Invoke ModelSim and check the libraries are mapped in the Library tab of the Workspace. If they are not mapped then it is easiest to map them from the command line using the `vmap` command e.g.:
`vmap unisim <complib_compile_location>/unisim`
7. Check you can run a simulation by loading one of the cores, either from the command line:
`vsim unisim.ppc405 -t ps`
 or from the GUI:
 - a. Go to Simulate > Start Simulation on the toolbar
 - b. Set the resolution to "ps"
 - c. Expand the unisim library
 - d. Highlight emac, gt or ppc405
 - e. Press OK.
8. If you get to the `VSIM>` prompt without errors then you are ready to go.

DISCLAIMER

This technical note is provided in good faith, and is intended to offer some guidance and ideas as to how you might improve the effectiveness with which you use our tools in the implementation of your designs. Whilst we make every endeavour to ensure the correctness of the information herein you should verify its suitability for use in your design as Saros Technology can accept no liability for any damages arising from its use howsoever caused.

3 SETTINGS FOR THE modelsim.ini FILE

The following settings need to be used for running the Swift Interface on a Windows platform. The important settings are marked in bold. All other lines should be commented out with a “;”.

```
[lmc]
; ModelSim's interface to Logic Modeling's SmartModel SWIFT software
; libsm = $MODEL_TECH/libsm.sl
; ModelSim's interface to Logic Modeling's SmartModel SWIFT software (Windows NT)
libsm = $MODEL_TECH/libsm.dll
; Logic Modeling's SmartModel SWIFT software (HP 9000 Series 700)
; libswift = $LMC_HOME/lib/hp700.lib/libswift.sl
; Logic Modeling's SmartModel SWIFT software (IBM RISC System/6000)
; libswift = $LMC_HOME/lib/ibmrs.lib/swift.o
; Logic Modeling's SmartModel SWIFT software (Sun4 Solaris)
; libswift = $LMC_HOME/lib/sun4Solaris.lib/libswift.so
; Logic Modeling's SmartModel SWIFT software (Windows NT)
libswift = $LMC_HOME/lib/pcnt.lib/libswift.dll
; Logic Modeling's SmartModel SWIFT software (Linux)
; libswift = $LMC_HOME/lib/x86_linux.lib/libswift.so

; ModelSim's interface to Logic Modeling's hardware modeler SFI software
libhm = $MODEL_TECH/libhm.sl
; ModelSim's interface to Logic Modeling's hardware modeler SFI software (Windows NT)
; libhm = $MODEL_TECH/libhm.dll
; Logic Modeling's hardware modeler SFI software (HP 9000 Series 700)
; libsfi = <sfi_dir>/lib/hp700/libsfi.sl
; Logic Modeling's hardware modeler SFI software (IBM RISC System/6000)
; libsfi = <sfi_dir>/lib/rs6000/libsfi.a
; Logic Modeling's hardware modeler SFI software (Sun4 Solaris)
; libsfi = <sfi_dir>/lib/sun4.solaris/libsfi.so
; Logic Modeling's hardware modeler SFI software (Windows NT)
; libsfi = <sfi_dir>/lib/pcnt/lm_sfi.dll
; Logic Modeling's hardware modeler SFI software (Linux)
```

Note: The MODEL_TECH variable is set automatically by Modelsim and MUST NOT be set by the user.

4 FURTHER READING

There is a detailed section on the Swift Interface in the ModelSim User Manual if you fancy some further reading. For Version 6.0x this can be found in Appendices F and G.

DISCLAIMER

This technical note is provided in good faith, and is intended to offer some guidance and ideas as to how you might improve the effectiveness with which you use our tools in the implementation of your designs. Whilst we make every endeavour to ensure the correctness of the information herein you should verify its suitability for use in your design as Saros Technology can accept no liability for any damages arising from its use howsoever caused.