@echo off

setlocal EnableDelayedExpansion

rem These need to change, for each new release. You might need to change

rem the SetParameters function, if the compiler changes. Otherwise, the

rem rest of the script should not need changing.

rem

set "revn=195"

set "revndot=19.5"

set "ROOT\_DIR=%AWP\_ROOT195%"

set "IFORT\_HOME=%IFORT\_COMPILER17%"

set "VSVER=vs2017"

rem :::::::::::::::::::::::::::::::

rem :: ::

rem :: E X E C U T I V E ::

rem :: ::

rem :::::::::::::::::::::::::::::::

call :CheckEnvVars || goto :MyExit

call :SetPlatformTarget || goto :MyExit

call :SetParameters || goto :MyExit

call :SetCompilerArgs || goto :MyExit

call :CompileSourceFiles || goto :MyExit

call :BuildCommonDll || goto :MyExit

call :BuildOtherDll || goto :MyExit

endlocal

goto :EOF

rem :::::::::::::::::::::::::::::::

rem :: ::

rem :: F U N C T I O N S ::

rem :: ::

rem :::::::::::::::::::::::::::::::

:CheckEnvVars

rem check some basic environment variables

rem

set "TXT1="

set "TXT2="

if not defined AWP\_ROOT195 (

set "TXT1= I'm sorry, but environment variable AWP\_ROOT%revn% does not exist."

) else if "%AWP\_ROOT195%"=="" (

set "TXT1= I'm sorry, but environment variable AWP\_ROOT%revn% exists, but is not set."

) else if not exist "%AWP\_ROOT195%\ansys\bin\winx64\ansys.exe" (

set "TXT1= I'm sorry, but environment variable AWP\_ROOT%revn% exists, and is set,"

set "TXT2= but it does not seem to point to a useful location."

)

if not "%TXT1%"=="" (

echo.

echo. %TXT1%

if not "%TXT2%"=="" ( echo. %TXT2% )

echo. It should be set to the install location, of the ANSYS software. For

echo. example:

echo.

echo. C:\Program Files\ANSYS Inc\v%revn%

echo.

echo. Please create/fix this variable, and then try again.

echo.

endlocal

exit /B 1

)

exit /B 0

rem Under normal circumstances, nothing below this line needs to change,

rem to revision this file.

rem

:SetPlatformTarget

if "%PROCESSOR\_ARCHITECTURE%"=="AMD64" (

set "PLATFORM\_DIR=winx64"

set "IFORT\_PLATFORM=intel64"

set "MACHINE\_TARGET=X64"

) else if "%PROCESSOR\_ARCHITEW6432%"=="AMD64" (

set "PLATFORM\_DIR=winx64"

set "IFORT\_PLATFORM=intel64"

set "MACHINE\_TARGET=X64"

) else if "%PROCESSOR\_ARCHITECTURE%"=="x86" (

set "PLATFORM\_DIR=intel"

set "IFORT\_PLATFORM=ia32"

set "MACHINE\_TARGET=X86"

)

exit /B 0

:SetParameters

rem "PLATFORM\_DIR=winx64"

rem "IFORT\_PLATFORM=intel64"

set "BIT\_TARGET=64"

rem We run the Intel-provided setup script, to ensure that the

rem build tools can find all the programs they need.

rem

call "%IFORT\_HOME%\bin\compilervars.bat" %IFORT\_PLATFORM% %VSVER%

rem Make sure we also use the M-APDL headers and libs

set "INCLUDE=%ROOT\_DIR%\ansys\customize\Include;%INCLUDE%"

set "LIB=%ROOT\_DIR%\ansys\Custom\Lib\%PLATFORM\_DIR%;%LIB%"

exit /B 0

:SetCompilerArgs

rem command-line arguments, for the compiler lines

rem

rem common macros, C macros, Fortran macros, 64-bit macros

set "COMMACS=/DNOSTDCALL /DARGTRAIL /DCADOE\_ANSYS /DPCWINNT\_SYS"

set "CMACS=/DCURVEFIT\_EXPORTS /D\_X86=1 /DOS\_WIN32 /DWIN32 /D\_\_STDC\_\_"

set "FMACS=/D\_EFL /DFORTRAN"

set "MACS64=/DPCWIN64\_SYS /DPCWINX64\_SYS"

rem common switches, C switches, Fortran switches

set "COMSWITCH=/O2 /MD /c"

set "CSWITCH=/Gy- /EHsc /Zi /W3"

set "FSWITCH=/fpp /4Yportlib /auto /Fo.\ /watch:source"

exit /B 0

:CompileSourceFiles

rem Yes, we just compile all of the source files we see, in this

rem working directory. This is done, because we never architected

rem a way, for the user to specify which source files go with which

rem DLLs.

rem

del /q compile.log compile\_error.txt >NUL 2>&1

del /q link.log link\_error.txt >NUL 2>&1

if "%PLATFORM\_DIR%"=="winx64" (

if exist \*.F ( ifort %COMMACS% %FMACS% %COMSWITCH% %FSWITCH% %MACS64% \*.F >>compile.log 2>&1 )

if exist \*.c ( cl %COMMACS% %CMACS% %COMSWITCH% %CSWITCH% %MACS64% \*.c >>compile.log 2>&1 )

if exist \*.cpp ( cl %COMMACS% %CMACS% %COMSWITCH% %CSWITCH% %MACS64% \*.cpp >>compile.log 2>&1 )

)

if "%PLATFORM\_DIR%"=="intel" (

if exist \*.F ( ifort %COMMACS% %FMACS% %COMSWITCH% %FSWITCH% /align:rec4byte \*.F >>compile.log 2>&1 )

if exist \*.c ( cl %COMMACS% %CMACS% %COMSWITCH% %CSWITCH% /Zp4 \*.c >>compile.log 2>&1 )

if exist \*.cpp ( cl %COMMACS% %CMACS% %COMSWITCH% %CSWITCH% /Zp4 \*.cpp >>compile.log 2>&1 )

)

if exist compile.log (

FINDSTR /I /C:": error" compile.log >compile\_error.txt

if !ERRORLEVEL!==0 (

call :CompileFail

exit /B 1

)

del /Q compile\_error.txt

)

exit /B 0

rem We have a feature, that implements a "common block" data storage area,

rem for use by the DLLs. If you are making use of this, then this must be

rem built first, so it can be linked with the other DLLs.

rem

:BuildCommonDll

rem We work from a list file, telling what DLL stuff to build. The list

rem is normally created by ansysNNN.exe, when it scans the principal input

rem file, looking for /UPF directives. We now scan that list, checking if

rem the common block DLL was specified. Bail, if not.

set "HIT=0"

for /f "eol= tokens=1 delims=. " %%U in (ansupf.lst) DO (

if /I "%%U" == "userdata" ( set "HIT=1" )

)

rem bail, if they aren't using the "common block" DLL feature

if "%HIT%" == "0" ( exit /B 0 )

rem bail, if the common block source file was not built

if not exist "userdata.F" ( exit /B 0 )

if not exist "userdata.obj" ( exit /B 0 )

rem we are, so clean-up, and prep for linking

call :AssembleLinkerFiles userdata

rem link it, into its' own DLL

link @%UPFFILE%.lrf >>link.log 2>&1

IF !ERRORLEVEL! GEQ 1 ( goto LinkFail )

rem get rid of this, so any further links won't use it

del /q userdata.obj >NUL 2>&1

exit /B 0

rem All other UPF DLLs are constructed here.

rem

:BuildOtherDll

set "BADHIT=0"

rem we only build files from a list

rem

for /f "eol= tokens=1 delims=. " %%U in (ansupf.lst) DO (

rem link everything except the common block

if /I not "%%U" == "userdata" (

rem clean-up, and prep for linking ...

call :AssembleLinkerFiles %%U

rem ... then link the DLL

echo. ======================================== >> link.log

echo. Linking %%U ... >> link.log

link @%%ULib.lrf >> link.log 2>&1

if !ERRORLEVEL! GEQ 1 (

call :ShowBanner "%%ULib.dll (%%U) has FAILED to build"

set "BADHIT=1"

) else (

call :ShowBanner "%%ULib.dll (%%U) has been successfully built."

)

echo. >> link.log

)

)

if "%BADHIT%" == "1" (

call :LinkFail

exit /B 1

)

rem set ANS\_USER\_PATH=%CD%

rem echo. \*\*\* ANS\_USER\_PATH: %ANS\_USER\_PATH% \*\*\*

rem get rid of these, or ansysNNN.exe will have a fit

del /q compile\_error.txt link\_error.txt >NUL 2>&1

exit /B 0

:AssembleLinkerFiles

rem clean old linker files, and create new ones

set "SRC=%1"

set "UPFFILE=%SRC%Lib"

del /q %UPFFILE%.lib %UPFFILE%.dll %UPFFILE%.lrf %UPFFILE%.map %UPFFILE%.def %UPFFILE%.exp >NUL 2>&1

rem The source file was already compiled, so just create

rem the exported-functions file, and the linker resource

rem file.

set "EXFILE=%UPFFILE%ex.def"

set "LRFFILE=%UPFFILE%.lrf"

if /I not "%SRC%" == "userdata" (

echo EXPORTS> %EXFILE%

echo. >> %EXFILE%

echo %SRC%>> %EXFILE%

"%ROOT\_DIR%\ansys\Custom\user\%PLATFORM\_DIR%\upcase" %EXFILE%

)

echo -out:%UPFFILE%.dll> %LRFFILE%

if /I not "%SRC%" == "userdata" (

echo -def:%EXFILE%>> %LRFFILE%

)

echo -dll>> %LRFFILE%

echo -machine:%MACHINE\_TARGET%>> %LRFFILE%

echo -map>> %LRFFILE%

echo -manifest:embed>> %LRFFILE%

echo -defaultlib:ANSYS.lib>> %LRFFILE%

if exist "userdataLib.lib" (

echo -defaultlib:userdataLib.lib>> %LRFFILE%

)

echo. >> %LRFFILE%

if "%SRC%" == "userdata" (

echo userdata.obj>> %LRFFILE%

) else (

echo \*.obj>> %LRFFILE%

)

exit /B 0

:CompileFail

call :ShowBanner "UPF COMPILER ERROR! Check compile.log for more information."

exit /B 0

:LinkFail

copy /y link.log link\_error.txt

call :ShowBanner "UPF LINK FAILED! Check link.log for more information"

exit /B 0

:ShowBanner

echo.

echo. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

echo.

echo. %1

echo.

echo. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

echo.

exit /B 0

:MyExit

endlocal

goto :eof