# Transcript Package User Guide

## User Guide for Release 2024.11

By

Jim Lewis

SynthWorks VHDL Training

Jim@SynthWorks.com

http://www.SynthWorks.com

## Table of Contents

## 1 TranscriptPkg Overview

TranscriptPkg allows different parts of a testbench to use a common transcript file. It does this by providing an internal file identifier (TranscriptFile), and subprograms for opening (TranscriptOpen) files, closing (TranscriptClose) files, printing (print and writeline), and checking if the file is open (IsTranscriptOpen).

## 2 TranscriptPkg Use Models

TranscriptPkg offers three printing methods, Print, WriteLine, and direct usage of TranscriptFile. If the TranscriptFile is open, Print and WriteLine write to the transcript file, otherwise, they write to std.textio.OUTPUT.

```
use osvvm.TranscriptPkg.all ;
architecture Test1 of tb is

  process
  begin
    TranscriptOpen("./results/Transcript_t1_nominal.txt") ;
    Print("Print 1") ;
    swrite(buf, "Write 1") ;
    writeline(buf) ;
    . . .
    TranscriptClose ;

    if IsTranscriptOpen then
      swrite(buf, "Write 1 - transcript open") ;
      writeline(TranscriptFile, buf) ;
    end if;
```

TranscriptPkg also supports a mirror mode in which Print and WriteLine write to both TranscriptFile and std.textio.OUTPUT. Entering mirror mode is via SetTranscriptMirror.

```
SetTranscriptMirror ;
SetAlertStopCount(ERROR, 20) ;
```

## 3 Usage by other OSVVM packages

Print and WriteLine are used by other OSVVM packages.

In CoveragePkg, WriteBin without a file specification uses IsTranscriptOpen to determine whether to use std.textio.OUTPUT or TranscriptFile.

In AlertLogPkg, alert, log, and ReportAlerts all use osvvm.TranscriptPkg.WriteLine.

## 4 Method Reference

### 4.1 Package Reference

Using TranscriptPkg requires the following package reference:

```
library osvvm ;
use osvvm.TranscriptPkg.all ;
```

### 4.2 Transcript File

TranscriptFile is an internal file to the declarative region of TranscriptPkg.

```
file TranscriptFile : text ;
```

### 4.3 TranscriptOpen

TranscriptOpen opens the TranscriptFile.  When the transcript is open, alert, log, ReportAlerts, WriteLine, and print use the TranscriptFile, otherwise, they use OUTPUT.

```
procedure TranscriptOpen (
  Status       : InOut FILE_OPEN_STATUS;
  ExternalName : STRING;
  OpenKind     : FILE_OPEN_KIND := WRITE_MODE) ;
procedure TranscriptOpen (
  ExternalName : STRING;
  OpenKind     : FILE_OPEN_KIND := WRITE_MODE) ;
impure function  TranscriptOpen (
  ExternalName : STRING;
  OpenKind     : FILE_OPEN_KIND := WRITE_MODE) return FILE_OPEN_STATUS ;

-- In ReportPkg.vhd
procedure TranscriptOpen (OpenKind: WRITE_APPEND_OPEN_KIND := WRITE_MODE) ;
procedure TranscriptOpen (
  Status       : InOut FILE_OPEN_STATUS;
  OpenKind     : WRITE_APPEND_OPEN_KIND := WRITE_MODE) ;
. . .
```

Usage of TranscriptOpen:

```
SetTestName("Test1") ;
TranscriptOpen ;   -- Opens Test1.log

-- Naming the log file
TranscriptOpen(Status,   "./Test2.log", WRITE_MODE) ;
TranscriptOpen(          "./Test3.log", WRITE_MODE) ;
```

If TranscriptOpen does not specify a file name (via parameter ExternalName), then the file will be named <TestName>.log where <TestName> is the name set by SetTestName.  If the OpenKind is not specified, WRITE_MODE is used.

Note that if you use OSVVM scripts, the scripts will move the transcript file to the directory ${OutputBaseDirectory}/${ResultsSubdirectory}/${TestSuiteName}. Hence, there is no need to include a path with the file.

Usage of the function form allows a file to be opened in a declarative region (such as in a package).

```
Constant OPENED_RESULTS : FILE_OPEN_STATUS :=
      TranscriptOpen("./Results.txt", WRITE_MODE) ;
```

## 4.4 TranscriptClose

TranscriptClose closes the TranscriptFile.

```
  procedure TranscriptClose ;
```

## 4.5 IsTranscriptOpen

IsTranscriptOpen returns true when the TranscriptFile is open.

```
impure function IsTranscriptOpen return boolean ;
. . .
if IsTranscriptOpen then
  swrite(buf, "Write 1 - transcript open") ;
  writeline(TranscriptFile, buf) ;
end if;
```

## 4.6 WriteLine

Writeline writes the buffer to the TranscriptFile when it is open, otherwise, to OUTPUT.

```
procedure WriteLine(buf : inout line)  ;
. . .
swrite(buf, "A Message") ;
writeline(buf) ;
```

## 4.7 Print

Print writes the string to the TranscriptFile if it is open, otherwise, to OUTPUT.

```
procedure Print(s : string) ;
. . .
Print("Another Message") ;
```

## 4.8 BlankLine

BlankLine puts a variable number of blank lines in the output by calling print("") ;.

```
procedure BlankLine (count : integer := 1) ;
. . .
BlankLine(5) ; -- prints 5 blank lines
. . .
```

### 4.9 SetTranscriptMirror

SetTranscriptMirror sets mirror mode in which Print and WriteLine write to both TranscriptFile and std.textio.OUTPUT.

```
procedure SetTranscriptMirror (A : boolean := TRUE) ;
. . .
SetTranscriptMirror ;
```

### 4.10 IsTranscriptMirrored

IsTranscriptMirrored returns true when the transcript is mirrored.

```
impure function  IsTranscriptMirrored return boolean ;
. . .
if IsTranscriptMirrored then
  . . .
```

## 5 About TranscriptPkg

TranscriptPkg is part of the OSVVM Utility library.

TranscriptPkg was created by Jim Lewis of SynthWorks.  Please support our work by buying your VHDL Training from SynthWorks.

TranscriptPkg.vhd is a work in progress and will be updated from time to time. Caution, undocumented items are experimental and may be removed in a future version.

## 6 Compiling and Using OSVVM Utility Library

Reference all packages in the OSVVM Utility library by using the context declaration:

```
library OSVVM ;
  context osvvm.OsvvmContext ;
```

Compilation order for OSVVM Utility Library is in OSVVM_release_notes.pdf.  Rather than learning this, we recommend using the OSVVM compilation scripts.

OSVVM Utility library is released under the Apache open source license.  It is free (both to download and use - there are no license fees).  You can download it from osvvm.org or from our development area on GitHub (https://github.com/OSVVM/OSVVM).

If you add features to the package, please donate them back under the same license as candidates to be added to the standard version of the package.  If you need features, be sure to contact us.

We also support the OSVVM user community and blogs through http://www.osvvm.org. Interested in sharing about your experience using OSVVM? Let us know, you can blog about it at osvvm.org.

## 7 About the Author - Jim Lewis

Jim Lewis, the founder of SynthWorks, has thirty plus years of design, teaching, and problem solving experience. In addition to working as a Principal Trainer for SynthWorks, Mr Lewis has done ASIC and FPGA design, custom model development, and consulting.

Mr. Lewis is chair of the IEEE 1076 VHDL Working Group (VASG) and is the primary developer of the Open Source VHDL Verification Methodology (OSVVM.org) packages. Neither of these activities generate revenue.

Please support our volunteer efforts by buying your VHDL training from SynthWorks.

If you find bugs these packages or would like to request enhancements, you can reach me at jim@synthworks.com.