

The Open Group Test Suites for POSIX[®] Realtime

Technical Data Sheet

March, 2001

The Open Group has introduced three new test suites for the latest IEEE POSIX[®] Realtime amendments. This is an opportunity for leading real-time suppliers to acquire state-of-the-art test tools to ensure the development of conformant POSIX[®] Realtime products.

The Open Group is well established as the premier open systems test supplier and certification authority. As a supplier independent and product neutral organization, The Open Group is the logical choice in developing standards-based test suites and certification schemes.

1. Introduction

The Open Group has expanded its portfolio of test suites that cover the complete POSIX 1003.1-1996 standard, including POSIX 1003.1b (realtime) and POSIX 1003.1c (threads) to include the latest POSIX Realtime amendments (1003.1d, 1003.1j and 1003.1q).

Use of test suites is essential for proper development and maintenance of standards-based products, ensuring conformance of supplier products to industry-standard APIs, application portability and interoperability. In-depth testing with Open Group test suites identifies defects at the earliest possible point in the development and test cycle minimizing costly defects found in the field.

2. The POSIX Realtime Amendments

This update to provide test suites for the POSIX Realtime Amendments includes tests for

- IEEE POSIX 1003.1d-1999 Additional Realtime Extension
- IEEE POSIX 1003.1j-200x Advanced Realtime Extension
- IEEE POSIX 1003.1q-200x Trace

The test suites for the POSIX Realtime amendments (.1d, .1j and .1q) have been developed as test packages under the Test Environment Toolkit and the VSXgen test framework, allowing them to be run stand-alone, or combined with other relevant VSXgen test packages.

3. VSRTE - POSIX 1003.1d Test Suite

3.1 Positioning

The Open Group Test Suite for the POSIX Additional Realtime Extensions is known as VSRTE. It is a standalone test suite covering POSIX 1003.1d and any associated Technical Corrigenda.

3.2 Functional Areas Tested

The test suite covers the following functional areas in the IEEE POSIX 1003.1d standard.

Spawn a Process

New system services to spawn the execution of a new process in an efficient manner.

Timeouts for some blocking services

Additional services that provide a timeout capability to system services already defined in POSIX.1b and POSIX.1c, thus allowing the application to include better error detection and recovery capabilities.

Sporadic Server Scheduling

The addition of a new scheduling policy appropriate for scheduling aperiodic processes or threads in hard real-time applications.

Execution Time Clocks and Timers

The addition of new clocks that measure the execution times of processes or threads, and the possibility to create timers based upon these clocks, for runtime detection (and treatment) of execution time overruns.

Advisory Information for File Management

Addition of services that allow the application to specify advisory information that can be used by the system to achieve better or even deterministic response times in file management or input & output operations.

3.3 POSIX Functions Tested

The following new POSIX functions will be tested:

posix_spawn()	posix_spawnattr_setsigdefault()
posix_spawn_file_actions_addclose()	posix_spawnattr_setsigmask()
posix_spawn_file_actions_adddup2()	posix_spawnnp()
posix_spawn_file_actions_addopen()	
posix_spawn_file_actions_destroy()	pthread_mutex_timedlock()
posix_spawn_file_actions_init()	sem_timedwait()
posix_spawnattr_destroy()	
posix_spawnattr_getflags()	mq_timedreceive()
posix_spawnattr_getpgroup()	mq_timedsend()
posix_spawnattr_getschedparam()	
posix_spawnattr_getschedpolicy()	clock_getcpuclockid()
posix_spawnattr_getsigdefault()	pthread_getcpuclockid()
posix_spawnattr_getsigmask()	
posix_spawnattr_init ()	posix_fadvise()
posix_spawnattr_setflags()	posix_fallocate()
posix_spawnattr_setpgroup()	posix_madvise()
posix_spawnattr_setschedparam()	posix_memalign()
posix_spawnattr_setschedpolicy()	

Tests are also performed on existing functions POSIX .1 functions whose behaviour is modified by the additional realtime extensions.

3.4 Header Files Tested

The following header files are tested:

<fcntl.h>

<limits.h>

<sys/mman.h>

<queue.h>	<semaphore.h>	<time.h>
<pthread.h>	<spawn.h>	<unistd.h>
<sched.h>	<stdlib.h>	

4. VSART - POSIX 1003.1j Test Suite

4.1 Positioning

The Open Group Test Suite for the POSIX Advanced Realtime Extensions is known as VSART. It is a standalone test suite covering POSIX 1003.1j and any associated Technical Corrigenda.

4.2 Functional Areas Tested

The test suite covers the following functional areas in the IEEE POSIX 1003.1j standard.

Memory management

A facility to allow programs to allocate or access different kinds of physical memory that are present in the system, and allow separate application programs to share portions of this memory.

Synchronization

Synchronization primitives that allow multiprocessor applications to achieve the performance benefits of their hardware architecture.

Clocks and Timers

The addition of the Monotonic Clock, the specification of the effects of setting the time of a clock on other timing services, and the addition of functions to support relative or absolute suspension based upon a clock specified by the application.

4.3 POSIX Functions Tested

The following new POSIX functions are tested:

posix_typed_mem_open()	pthread_barrier_init()
posix_mem_offset ()	pthread_barrier_destroy()
posix_typed_mem_get_info()	pthread_barrier_wait()
	thread_rwlockattr_init()
pthread_spin_init()	pthread_rwlockattr_destroy()
pthread_spin_destroy()	pthread_rwlockattr_getpshared()
thread_spin_lock()	pthread_rwlockattr_setpshared()
pthread_spin_trylock()	pthread_rwlock_init()
pthread_spin_unlock()	pthread_rwlock_destroy()
pthread_barrierattr_init()	pthread_rwlock_rdlock()
pthread_barrierattr_destroy()	pthread_rwlock_tryrdlock()
pthread_barrierattr_getpshared()	pthread_rwlock_timedrdlock()
pthread_barrierattr_setpshared()	pthread_rwlock_wrlock()

pthread_rwlock_trywrlock()	clock_nanosleep()
pthread_rwlock_timedwrlock()	pthread_condattr_setclock()
pthread_rwlock_unlock()	pthread_condattr_getclock()

Tests are also performed on existing functions POSIX .1 functions whose behaviour is modified by the advanced realtime functionality.

4.4 Header Files Tested

The following header files are - tested:

<sys/mman.h>	<stat.h>	<sys/types.h>
<pthread.h>	<time.h>	<unistd.h>

5. VSTRC - POSIX 1003.1q Test Suite

5.1 Positioning

The Open Group Test Suite for the POSIX Tracing amendment is known as VSTRC. It is a standalone test suite covering POSIX 1003.1q and any associated Technical Corrigenda.

5.2 Functional Areas Tested

The test suite covers the following functional areas in the IEEE POSIX 1003.1q standard.

5.3 POSIX Functions Tested

The following new POSIX functions are tested:

pthread_attr_gettracingstate()	posix_trace_attr_getmaxsystemevent size()
pthread_attr_settracingstate()	posix_trace_attr_getname()
pthread_gettracingstate()	posix_trace_attr_getstreamfullpolicy()
pthread_settracingstate()	posix_trace_attr_getusereventsize()
posix_trace_attr_destroy()	posix_trace_attr_getutsname()
posix_trace_attr_getclockres()	posix_trace_attr_init()
posix_trace_attr_getcreatetime()	posix_trace_attr_setinherited()
posix_trace_attr_getgenversion()	posix_trace_attr_setlogfullpolicy()
posix_trace_attr_getinherited()	posix_trace_attr_setmaxdatasize()
posix_trace_attr_getlogfullpolicy()	posix_trace_attr_setlogsize()
posix_trace_attr_getmaxdatasize()	posix_trace_attr_setstreamsize()
posix_trace_attr_getlogsize()	posix_trace_attr_setname()
posix_trace_attr_getstreamsize()	

posix_trace_attr_setstreamfullpolicy(
)
posix_trace_close()
posix_trace_create()
posix_trace_create_withlog()
posix_trace_eventid_open()
posix_trace_eventid_equal()
posix_trace_trid_eventid_open()
posix_trace_eventid_get_name()
posix_trace_eventtypelist_getnext_id
(
)
posix_trace_eventtypelist_rewind()
posix_trace_eventset_add()
posix_trace_eventset_del()
posix_trace_eventset_empty()
posix_trace_eventset_fill()
posix_trace_eventset_complement()
posix_trace_eventset_ismember()
posix_trace_flush()
posix_trace_get_attr()
posix_trace_get_filter()
posix_trace_get_status()
posix_trace_getnext_event()
posix_trace_open()
posix_trace_rewind()
posix_trace_set_filter()
posix_trace_shutdown()
posix_trace_start()
posix_trace_stop()
posix_trace_clear()
posix_trace_timedgetnext_event()
posix_trace_trygetnext_event()
posix_trace_event()

Tests are also performed on existing functions POSIX .1 functions whose behaviour is modified by the trace functionality.

5.4 Header Files Tested

The following header files are tested:

<pthread.h>

<trace.h>

6. The Test Environment Toolkit (TET)

6.1 TET Overview

The Test Environment Toolkit (TET) is a multi-platform uniform test scaffold, into which non-distributed and distributed test suites can be incorporated. It allows production of test suites sharing a common interface, therefore promoting sharing of test suites within organizations as well as between different organizations. Standardization of the test methodology and tools allows testing efforts to focus away from the harness and tools, thus increasing efficiency and productivity.

TET provides facilities to execute test cases in several ways as follows:

- Execution of sequences of test cases in a repeatable manner
- Execution of a single test case selected at random from a list of test cases
- Execution of test cases in parallel

Sequences of the above elements executing a specified number of times or until some time period has expired.

7. The VSXgen Framework

7.1 VSXgen Overview

VSXgen is an operating system test framework that allows tests for standard operating system APIs to be abstracted away from the framework itself. VSXgen builds upon the Test Environment Toolkit. This permits a common look and feel to test suites that are developed to reuse the framework, reduced development time, the ability to run the tests standalone as individual test suites, as well as to integrate them into a meta test suite.

For users familiar with installing and running one VSXgen based test suite, there is a minimal learning curve to running another test suite based on VSXgen. The framework itself is based on the core of the X/Open test suite VSX4, and is proven with over 10 years of use in the industry.

7.2 Current test subsets using VSXgen

The test suites supported as test subsets under the VSXgen structure are (as of March 2001):

- VSRT - Realtime (1003.1b/.1i)
- VSTH - Threads (1003.1c/Aspen threads)
- VSX4 - The VSX4 test suite (POSIX.1/XPG4 extensions)
- VSX5 - The Large File, Dynamic Linking and ISO MSE tests
- VSU5 - The UNIX® extensions tests.

8. Pricing

There is an incentive for companies who elect to purchase the complete bundle of test suites. For those companies who sign up to join the Real-time and Embedded Systems Forum, the right-to-use these tests will be bundled.

Test Suite	Single Site		Up to 3 Sites	
	10 Yr End User	1 Yr Support	10 Yr End User	1 Yr Support
VS RTE	\$ 5k	\$ 5k	\$ 7k	\$ 7k
VS ART	\$ 5k	\$ 5k	\$ 7k	\$ 7k
VS TRC	\$ 5k	\$ 5k	\$ 7k	\$ 7k
Bundled Price	\$ 14K	\$ 14k	\$ 20K	\$ 20k
RT Forum member	Right-to-use	\$ 14k	Right-to-use	\$ 20k

9. More Information

For more information, please visit our Web site at www.opengroup.org/testing/, or contact:

Paul Hickey

Region Manager
North America West,
Asia Pacific
+1 650 323 7992 x 240
<mailto:p.hickey@opengroup.org>

Birgit Hartje

Region Manager
North America East,
Central & South America
+1 781 376 8208
<mailto:b.hartje@opengroup.org>

Chris Parnell

Region Manager
Europe, Middle East, Africa
+44 (0) 118 950 8311 x2270
<mailto:c.parnell@opengroup.org>

Or send email to sales-team@opengroup.org.