Product Standard

Operating System and Languages:

Multi-Purpose Realtime Operating System

The Open Group

Product Standard

Operating System and Languages: Multi-Purpose Realtime Operating System

Document Number: X98RT

Published in the U.K. by The Open Group, July 1998.

Any comments relating to the material contained in this document may be submitted to:

The Open Group
Apex Plaza
Forbury Road
Reading
Berkshire RG1 1AX
U.K.

Or by email to:

OGSpecs@opengroup.org

Product Standard

NAME

Multi-Purpose Realtime Operating System

LABEL FOR LOGO

No label.

DESCRIPTION

This Product Standard defines interfaces and functionality to support the source portability of applications with realtime requirements. This Product Standard is aligned with ANSI/IEEE Std. 1003.1b-1993, ANSI/IEEE Std. 1003.1c-1995, and ANSI/IEEE Std. 1003.1i-1995.[1]

The definition of realtime is as follows:

- Realtime in operating systems: the ability of the operating system to provide a required level of service in a bounded response time.

The specific mandatory functional areas covered by this Product Standard are:

- Semaphores (ISO/IEC 9945-1: 1996 (POSIX-1), clause 11)
- Process memory locking (ISO/IEC 9945-1: 1996 (POSIX-1), clause 2.1)
- Memory mapped files and shared memory objects (ISO/IEC 9945-1: 1996 (POSIX-1), clause 12.2 and 12.3)
- File synchronization (ISO/IEC 9945-1: 1996 (POSIX-1), clause 6)
- Priority scheduling (ISO/IEC 9945-1: 1996 (POSIX-1), clause 13)
- Realtime signals (ISO/IEC 9945-1: 1996 (POSIX-1), clause 3)
- Timers (ISO/IEC 9945-1: 1996 (POSIX-1), clause 14)
- Synchronized input and output (ISO/IEC 9945-1: 1996 (POSIX-1), clause 5 and 6)
- Asynchronous input and output (ISO/IEC 9945-1: 1996 (POSIX-1), clause 6)

One optional area is defined:

- Threads (ISO/IEC 9945-1: 1996 (POSIX-1), clause 13, 16, 17, and 18)

1. These are incorporated in ISO/IEC 9945-1: 1996 (POSIX-1): ISO/IEC 9945-1: 1996, Information Technology — Portable Operating System Interface (POSIX) — Part 1: System Application Program Interface (API) [C Language] (identical to ANSI/IEEE Std. 1003.1-1996). Incorporating ANSI/IEEE Std. 1003.1-1990, 1003.1b-1993 (Realtime), 1003.1c-1995 (Threads), and 1003.1i-1995 (also Realtime).

Threads includes functionality to support multiple flows of control within a process. The specific functional areas covered by threads and their scope include:

- Thread management: the creation, control, and termination of multiple flows of control in the same process under the assumption of a common shared address space.
- Synchronization primitives optimized for tightly coupled operation of multiple control flows in a common, shared address space.

The Threads interfaces are fully aligned with ANSI/IEEE Std. 1003.1c-1995. Within the Threads option is a single sub-option:

- The Realtime Threads Feature Group, as denoted by the RTT shaded extension in System Interfaces and Headers, Issue 5,[2] Section 2.8, Threads.

CONFORMANCE REQUIREMENTS

Human-Computer Interface

System Interface Definitions, Issue 5,[3] Section 4.1, Portable Character Set, Table 4-1, glyphs.

Portability Interface

The following system interfaces and headers listed in System Interfaces and Headers, Issue 5 (excluding any EX extensions marked) are mandatory:

aio_cancel( )
aio_error( )
aio_fsync( )
aio_read( )
aio_return( )
aio_suspend( )
aio_write( )
clock_getres( )
clock_gettime( )
clock_settime( )
fchmod( )
fdatasync( )
fsync( )
ftruncate( )
lio_listio( )
mlockall( )
mmap( )
mprotect( )
mq_close( )
mq_getattr( )
mq_notify( )
mq_open( )
mq_receive( )
mq_send( )
mq_setattr( )
mq_unlink( )
msync( )
munlockall( )
munmap( )
nanosleep( )
sched_get_priority_max( )
sched_get_priority_min( )
sched_getparam( )
sched_getscheduler( )
sched_rr_get_interval( )
sched_setparam( )
sched_setscheduler( )
sched_yield( )
sem_close( )
sem_destroy( )
sem_getvalue( )
sem_init( )
sem_open( )
sem_post( )
sem_trywait( )
sem_unlink( )
sem_wait( )
shm_open( )
shm_unlink( )
sigqueue( )
sigtimedwait( )
sigwaitinfo( )
timer_create( )
timer_delete( )
timer_getoverrun( )
timer_gettime( )
timer_settime( )

<aio.h>
<mqueue.h>
<sched.h>
<semaphore.h>
<sys/mman.h>

---

2. CAE Specification, January 1997, System Interfaces and Headers, Issue 5 (ISBN: 1-85912-181-0, C606), published by The Open Group.
3. CAE Specification, January 1997, System Interface Definitions, Issue 5 (ISBN: 1-85912-186-1, C605), published by The Open Group.

In addition, semantics of the following system interfaces and headers marked by the RT extension are mandatory:

close( )	*fdopen*( )	*read*( )
exec( )	*fork*( )	*sigaction*( )
exit( )	*fstat*( )	*sigwait*( )
fchmod( )	*lseek*( )	*sysconf*( )
fcntl( )	*open*( )	*write*( )
<errno.h>	**<signal.h>**	**<time.h>**
<fcntl.h>	**<sys/mman.h>**	**<unistd.h>**
<limits.h>	**<sys/types.h>**	

If the Threads option within this Product Standard is supported, then the following symbols are always defined:

_POSIX_THREADS
_POSIX_THREAD_ATTR_STACKADDR
_POSIX_THREAD_ATTR_STACKSIZE
_POSIX_THREAD_PROCESS_SHARED

and the following system interfaces listed in System Interfaces and Headers, Issue 5, Section 2.8, Threads shall all be supported:

pthread_atfork( )	*pthread_detach*( )
pthread_attr_destroy( )	*pthread_equal*( )
pthread_attr_getdetachstate( )	*pthread_exit*( )
pthread_attr_getschedparam( )	*pthread_getspecific*( )
pthread_attr_getstackaddr( )	*pthread_join*( )
pthread_attr_getstacksize( )	*pthread_key_create*( )
pthread_attr_init( )	*pthread_key_delete*( )
pthread_attr_setdetachstate( )	*pthread_kill*( )
pthread_attr_setschedparam( )	*pthread_mutex_destroy*( )
pthread_attr_setstackaddr( )	*pthread_mutex_init*( )
pthread_attr_setstacksize( )	*pthread_mutex_lock*( )
pthread_cancel( )	*pthread_mutex_trylock*( )
pthread_cleanup_pop( )	*pthread_mutex_unlock*( )
pthread_cleanup_push( )	*pthread_mutexattr_destroy*( )
pthread_cond_broadcast( )	*pthread_mutexattr_getpshared*( )
pthread_cond_destroy( )	*pthread_mutexattr_init*( )
pthread_cond_init( )	*pthread_mutexattr_setpshared*( )
pthread_cond_signal( )	*pthread_once*( )
pthread_cond_timedwait( )	*pthread_self*( )
pthread_cond_wait( )	*pthread_setcancelstate*( )
pthread_condattr_destroy( )	*pthread_setcanceltype*( )
pthread_condattr_getpshared( )	*pthread_setspecific*( )
pthread_condattr_init( )	*pthread_sigmask*( )
pthread_condattr_setpshared( )	*pthread_testcancel*( )
pthread_create( )	*sigwait*( )

In addition, if the Threads option within this Product Standard is supported, the following symbol is always defined:

_POSIX_THREAD_SAFE_FUNCTIONS

and the following system interfaces listed in System Interfaces and Headers, Issue 5, Section 2.8, Threads shall all be supported:

asctime_r( )	*getpwnam_r*( )
ctime_r( )	*getpwuid_r*( )
flockfile( )	*gmtime_r*( )
ftrylockfile( )	*localtime_r*( )
funlockfile( )	*putc_unlocked*( )
getc_unlocked( )	*putchar_unlocked*( )
getchar_unlocked( )	*rand_r*( )
getgrgid_r( )	*readdir_r*( )
getgrnam_r( )	*strtok_r*( )

If the Threads option within this Product Standard is supported, the following system interfaces listed in System Interfaces and Headers, Issue 5, Section 2.8, Threads, as marked by the RTT shaded extension, are optional and may be supported if the Realtime Threads Feature Group is supported:

pthread_attr_getinheritsched( )	*pthread_mutex_getprioceiling*( )
pthread_attr_getschedpolicy( )	*pthread_mutex_setprioceiling*( )
pthread_attr_getscope( )	*pthread_mutexattr_getprioceiling*( )
pthread_attr_setinheritsched( )	*pthread_mutexattr_getprotocol*( )
pthread_attr_setschedpolicy( )	*pthread_mutexattr_setprioceiling*( )
pthread_attr_setscope( )	*pthread_mutexattr_setprotocol*( )
pthread_getschedparam( )	*pthread_setschedparam*( )

Programming Language Environment

C Language. Dialect ISO C. ISO C source programs invoking the services of this Product Standard must be supported by the registered product.

Interoperability

- Data Interchange Formats

  Not defined.

- Communications Interfaces and Protocols

  Not defined.

OPERATIONAL ENVIRONMENT

ISO/IEC 9945-1:1996 (POSIX-1).

PORTABILITY ENVIRONMENT

ISO/IEC 9945-1:1996 (POSIX-1).

OVERRIDING STANDARDS

ANSI/IEEE Std. 1003.1b-1993, ANSI/IEEE Std. 1003.1c-1995, ANSI/IEEE Std. 1003.1i-1995, incorporated in ISO/IEC 9945-1: 1996 (POSIX-1).

ISO/IEC 9899: 1990 (C Language).[4]

INDICATORS OF COMPLIANCE

A Test Report from the currently authorized release of the VSRT Test Suite.

A Test Report from the currently authorized release of the VSX4 Test Suite run in POSIX.1-1996 mode.

If the Threads option within this Product Standard is supported, a Test Report from the currently authorized release of the VSTH Test Suite.

For ISO C, any of the following:

- Current NIST certificate of ANSI/ISO C Language validation.
- Current entry in the NIST register of validated language processors for ANSI/ISO C Language indicating no non-conformities.
- Current Certificate of ANSI/ISO C Language validation issued by a member of the GLATC European testing agreement group registered under ECITC.
- A Test Report from a currently authorized release of either the Perennial ANSI C Validation Suite or the Plum-Hall C Validation Suite, together with the checklist of Implementation-defined Features.

MIGRATION

This Product Standard is primarily a subset of the Internationalized System Calls and Libraries Extended V2 Product Standard. As such it is new functionality, and there are very few incompatibility issues in migrating existing applications.

Detailed migration information can be found in Go Solo 2.[5]

4. ISO/IEC 9899:1990, Programming Languages — C (technically identical to ANSI standard X3.159-1989).
5. Go Solo 2, May 1997 (ISBN: 0-13-575689-8, X909P).