MapuSoft Technologies 1.3.8 Release Notes

Release 1.3.8
December, 2012
Revision 1

These release notes accompany Release 1.3.8 of MapuSoft Technologies. They briefly describe the software features and provide a summary of the current software limitations and known defects, if any, that exist in this release.

Contents

1.3.8 Release Updated Products

Release 1.3.8 of MapuSoft Technologies encompasses all upgrades to MapuSoft's OS Abstractor, OS Changer solutions. Also, added are two new products called Ada-C/C++ Changer and Ada-PAL Compiler. The Release 1.3.8 package components are as follows:

Table 1: Updated Products

No	Component Name	Version	Product
1	demo_ada_to_c	3.5.5.12440	Ada-C/C++ Changer
2	demo_cross_os	3.5.5.13405	OS Abstractor / OS Changer
3	demo_nucleus	3.5.5.13406	OS Abstractor / OS Changer
4	demo_posix	3.5.5.13407	OS Abstractor / OS Changer
5	demo_psos	3.5.5.13408	OS Abstractor / OS Changer
6	demo_psos_classic	3.5.5.12450	OS Abstractor / OS Changer
7	demo_uitron	3.5.5.13409	OS Abstractor / OS Changer
8	demo_vxworks	3.5.5.13410	OS Abstractor / OS Changer
9	demo_windosws	3.5.5.13411	OS Abstractor / OS Changer
10	cross_os_android	3.5.5.13490	OS Abstractor / OS Changer
11	cross_os_linux	3.5.5.13514	OS Abstractor / OS Changer
12	cross_os_lynxos	3.5.5.13492	OS Abstractor / OS Changer
13	cross_os_mqx	3.5.5.13493	OS Abstractor / OS Changer
14	cross_os_netbsd	3.5.5.13494	OS Abstractor / OS Changer
15	cross_os_nucleus	3.5.5.13495	OS Abstractor / OS Changer
16	cross_os_qnx	3.5.5.13496	OS Abstractor / OS Changer
17	cross_os_solaris	3.5.5.13497	OS Abstractor / OS Changer
18	cross_os_threadx	3.5.5.13498	OS Abstractor / OS Changer
19	cross_os_ucos	3.5.5.13499	OS Abstractor / OS Changer
20	cross_os_uitron	3.5.5.13500	OS Abstractor / OS Changer
21	cross_os_vxworks	3.5.5.13501	OS Abstractor / OS Changer
22	cross_os_windows	3.5.5.13513	OS Abstractor / OS Changer
23	include	3.5.5.12785	OS Abstractor / OS Changer
24	nucleus_interface	3.5.5.13450	OS Abstractor / OS Changer
25	posix_interface	3.5.5.13453	OS Abstractor / OS Changer
26	psos_classic_interface	3.5.5.12462	OS Abstractor / OS Changer
27	psos_interface	3.5.5.13451	OS Abstractor / OS Changer
28	uitron_interface	3.5.5.13452	OS Abstractor / OS Changer
29	vxworks_interface	3.5.5.13454	OS Abstractor / OS Changer
30	windows_interface	3.5.5.13418	OS Abstractor / OS Changer
31	APPCOE Ada Compiler	3.999d+	Ada-PAL Compiler
32	APPCOE Ada Changer	3.999d+	Ada-C/C++ Changer
33	APPCOE Profiler Engine	1.1	OS Abstractor / OS Changer
34	APPCOE	1.3.8	APPCOE

The following tools have been modified and are included in AppCOE release package under license CPL license from Eclipse foundation (http://www.eclipse.org/legal/cpl-v10.html):

Tool	Version
Eclipse	3.6 [Helios]
CDT Eclipse Plug-in	7.0
BIRT Eclipse Plug-in	2.6

Click here http://mapusoft.com/downloads/ to get a free Evaluation CD.
Click here http://mapusoft.com/products/techdata/ for the latest Porting and Abstraction Lab User Manual.

API Interfaces Supported on Host Platform

Applications can be developed under APPCOE host environment via the various OS Interface Simulators provided by APPCOE. The following are the API interfaces supported on various host platforms:

Table 2: API Interfaces Supported on Host Platforms

APPCOE Host Platform / OS	Cross-OS	POSIX	VxWorks®	µITRON	pSOS®	pSOS 1.4 [1] pSOS Classic	Nucleus®	Windows®
Linux®	√	√	√	√	√	√	√	√
Windows	√	√	√	√	√	√	√	

API Interfaces and Supported Target OS

MapuSoft Technologies now provides OS support to the following API interfaces:

Table 3: API Interfaces and Support Target OS

Target OS	Cross-OS	POSIX	VxWorks®	µITRON	pSOS®	pSOS 1.4 [1] pSOS Classic	Nucleus®	Windows
VxWorks® 6x/5x	√	√		√	√	√	√	√
Linux® 2.4/2.6	√	√	√	√	√	√	√	√
RT Linux	√	√	√	√	√	√	√	√
LynxOS®	√	√	√	√	√	√	√	√
LynxOS-SE®	√	√	√	√	√	√	√	√
Solaris®	√	√	√	√	√	√	√	√
Unix®	√	√	√	√	√	√	√	√
eCOS®	√	√	√	√	√	√	√	√
Windows® XP/Vista/CE/7	√	√	√	√	√	√	√	
Nucleus®	√	√	√	√	√	√		√
ThreadX®	√	√	√	√	√	√	√	√
MQX®	√	√	√	√	√	√	√	√
QNX®	√	√	√	√	√	√	√	√
T-Kernel®	√	√	√	√	√	√	√	√
µITRON	√	√	√	√	√	√	√	√
µC/OS-III	√	√	√	√	√	√	√	√
NetBSD	√	√	√	√	√	√	√	√
Android	√	√	√	√	√	√	√	√
Solaris	√	√	√	√	√	√	√	√
Note1: MapuSoft uses pSOS 1.4 Rev. 3/10/1986 (product called pSOS Classic)								

MapuSoft Feature Support Table

MapuSoft Technologies provides support to the following features:

Table 4: Feature Support Table

Feature	Android	NetBSD	Linux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	Windows XP/ Vista/Mobile	Windows CE	µITRON	µC/OS-III	VxWorks	T-Kernel	eCOS	Unix	RT Linux
Signaling	Y	Y	Y	Y	Y	Y	N	Y	N	Y	Y	Y	N	Y	Y	Y	Y	Y
Profiling	N	Y	Y	Y	Y	Y	N	N	N	Y	Y	N	N	Y	Y	Y	Y	Y
Process	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Task Pooling	Y	Y	Y	Y	Y	Y	Y	N	N	Y	Y	Y	Y	Y	Y	Y	Y	Y
SMP Safe Protection[2]	N	Y	Y	Y	Y	Y	N	N	N	Y	Y	N	N	Y	Y	Y	Y	Y
Dead Synchronization Monitor	N	N	N	N	Y	N	N	N	N	N	N	N	N	N	N	N	N	N
ANSI Memory Mapping	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ANSI Formatted I/O Mapping	Y[1]	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ANSI I/O Mapping	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

	Notes
	1: Android maps printf to /dev/null. OS_printf has been mapped to the Android logging facility. In order to have any functionality with printf, MAP_OS_ANSI_FMT_IO must be set to OS_TRUE **2**: SMP support has been added since 1.3.6.1 release. For more information refer to Table 2.

MapuSoft Supported Tool Sets

APPCOE target features can be utilized across a wide variety of tools/IDE. However, this release was tested on the following IDE/Tool sets.

Table 5: Tested Tool Sets

Target OS	IDE/Tools	OS Version	CPU
µItron	HEW 9.2	Toppers 1.1	SH
Android	make	Version 1.5	Arm
Linux	eclipse 3.6	kernel build 2.4	x86
		kernel build 2.6 or later	x86
	make	kernel build 2.4	x86
		kernel build 2.6 or later	x86
LynxOS	make	Version 4.0	x86
		Version 4.2	x86
MQX	Code Warrior	Version 10.2	x86
NetBSD	make	Version 6.0	x86
Nucleus	Visual Studio 6.0	Version 1.1.13	x86
QNX	Momentics 4.5	Version 6.4.1	x86
Solaris	make	Version 11	x86
		Version 10	Sparc
ThreadX	Visual Studio.NET 2008	Version 5.4	x86
µC/OS-III	Visual Studio 6.0	Version 2	x86
VxWorks	Workbench 2.6	6.4 kernel	x86
		6.4 kernel	xcal
		6.4 RTP	x86
		6.4 RTP	xcal
	Workbench 3.1	6.7 kernel	x86
		6.7 kernel	xcal
		6.7 RTP	x86
		6.7 RTP	xcal
Windows	eclipse 3.6	XP	x86
		Vista	x86
		Windows 7	x86
		Windows 8	x86
	Visual Studio 6.0	XP	x86
		Vista	x86
	Visual Studio.NET 2005	XP	x86
		Vista	x86
		Windows 7	x86
		Windows 8	x86

Target OS	IDE/Tools	OS Version	CPU
	Visual Studio.NET 2008	XP	x86
		Vista	x86
		Windows 7	x86
		Windows 8	x86
Windows CE	Visual Studio.NET 2005	Version 5.0	x86
	Visual Studio.NET 2008	Version 5.0	x86
Windows Mobile	Visual Studio.NET 2005	Version 6.0	x86
	Visual Studio.NET 2008	Version 6.0	x86

Windows Interface

New APIs—The following new APIs have been added:

- **OS_Adopt_Native_Thread_To_Windows_Interface() –** This API configures an adopted thread so it can use the Windows Interface APIs. The native thread must have been adopted by calling OS_Adopt_Native_Thread_To_Cross_OS() API prior to making this function call.

Behavioral Changes of APIs — The following APIs behavioral changes have been made:

- All Resource Create API's [Task,Sema,Mutex,Queue, Pipe…etc] are created as local [OS_SCOPE_PROCESS] by default instead of shared [OS_SCOPE_SYSTEM].

  If a customer wishes to use the added inter-process communication functionality that Cross OS provides, all they would have to do is add the appropriate API call into their code. If they do not want this, they don't have to do anything.

  Here are the list of Scope change API's:
  - **OS_Set_Queue_Scope_To_System(OS_QUEUE id)**
  - **OS_Set_Pipe_Scope_To_System(OS_PIPE id)**
  - **OS_Set_Mutex_Scope_To_System(OS_MUTEX id)**
  - **OS_Set_Event_Group_Scope_To_System(OS_EVENT_GROUP id)**
  - **OS_Set_Semaphore_Scope_To_System(OS_SEMAPHORE id)**

Removed APIs—The following APIs have been deleted:

- None

Renamed APIs —The following APIs have been renamed:

- None

Windows Interface APIs

The following table provides more information on Windows Interface API level of support across different target Oss.

Table 6: Windows Interface APIs

Windows API	Android	NetBSD	Linux	RTLinux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	WindowsXP/ Vista/Mobile	Windows CE	µITRON	µC/OS-III	VxWorks
Handles															
CloseHandle	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
DuplicateHandle	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
GetHandleInformation	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
SetHandleInformation	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
Memory															
CopyMemory	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
FillMemory	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
MoveMemory	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
SecureZeroMemory	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ZeroMemory	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Pipes															
CreatePipe	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
CallNamedPipe	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ConnectNamedPipe	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
CreateNamedPipe	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
DisconnectNamedPipe	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
GetNamedPipeClientComputerName	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
GetNamedPipeClientProcessId	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
GetNamedPipeClientSessionId	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
GetNamedPipeHandleState	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
GetNamedPipeInfo	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
GetNamedPipeServerProcessId	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
GetNamedPipeServerSessionId	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
PeekNamedPipe	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
SetNamedPipeHandleState	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
TransactNamedPipe	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N

Windows API	Android	NetBSD	Linux	RTLinux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	WindowsXP/ Vista/Mobile	Windows CE	µITRON	µC/OS-III	VxWorks
WaitNamedPipe	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ReadFile	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
WriteFile	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Non ANSI String															
CharLower	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
CharLowerBuff	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
CharNext	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
CharNextExA	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
CharPrev	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
CharPrevExA	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
CharToOem	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
CharToOemBuff	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
CharUpper	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
CharUpperBuff	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
CompareString	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
CompareStringEx	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
FoldString	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
GetStringTypeA	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
GetStringTypeEx	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
GetStringTypeW	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
IsCharAlpha	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
IsCharAlphaNumeric	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
IsCharLower	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
IsCharUpper	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
LoadString	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
lstrcat	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
lstrcmp	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
lstrcmpi	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
lstrcpy	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
lstrcpyn	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
lstrlen	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
OemToChar	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
OemToCharBuff	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
StringCbCat	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
StringCbCatEx	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
StringCbCatN	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Windows API	Android	NetBSD	Linux	RTLinux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	WindowsXP/ Vista/Mobile	Windows CE	µITRON	µC/OS-III	VxWorks
StringCbCatNEx	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
StringCbCopy	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
StringCbCopyEx	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
StringCbCopyN	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
StringCbCopyNEx	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
StringCbGets	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
StringCbGetsEx	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
StringCbLength	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
StringCbPrintf	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
StringCbPrintfEx	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
StringCbVPrintf	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
StringCbVPrintfEx	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
StringCchCat	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
StringCchCatEx	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
StringCchCatN	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
StringCchCatNEx	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
StringCchCopy	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
StringCchCopyEx	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
StringCchCopyN	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
StringCchCopyNEx	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
StringCchGets	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
StringCchGetsEx	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
StringCchLength	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
StringCchPrintf	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
StringCchPrintfEx	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
StringCchVPrintf	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
StringCchVPrintfEx	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
wsprintf	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
wvsprintf	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Critical Sections															
DeleteCriticalSection	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
EnterCriticalSection	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
InitializeCriticalSection	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
InitializeCriticalSectionAndSpinCount	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
InitializeCriticalSectionEx	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
LeaveCriticalSection	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
SetCriticalSectionSpinCou	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Windows API	Android	NetBSD	Linux	RTLinux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	WindowsXP/ Vista/Mobile	Windows CE	µITRON	µC/OS-III	VxWorks
nt															
TryEnterCriticalSection	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Events															
CreateEvent	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
CreateEventEx	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
OpenEvent	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
PulseEvent	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ResetEvent	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
SetEvent	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
CreateMutex	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
CreateMutexEx	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
OpenMutex	Y	Y	Y	Y	Y	Y	Y	N	Y	N	Y	Y	Y	N	Y
ReleaseMutex	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Semaphore															
CreateSemaphore	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
CreateSemaphoreEx	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
OpenSemaphore	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ReleaseSemaphore	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Linked Lists															
InitializeSListHead	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
InterlockedFlushSList	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
InterlockedPopEntrySList	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
InterlockedPushEntrySList	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
QueryDepthSList	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
RtlFirstEntrySList	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
RtlInitializeSListHead	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
RtlInterlockedFlushSList	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
RtlInterlockedPopEntrySList	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
RtlInterlockedPushEntrySList	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
RtlQueryDepthSList	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Windows API	Android	NetBSD	Linux	RTLinux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	WindowsXP/ Vista/Mobile	Windows CE	µITRON	µC/OS-III	VxWorks
Timer Queues															
ChangeTimerQueueTimer	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
CreateTimerQueue	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
CreateTimerQueueTimer	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
DeleteTimerQueue	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
DeleteTimerQueueEx	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
DeleteTimerQueueTimer	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Wait (Handles)															
MsgWaitForMultipleObjects	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
MsgWaitForMultipleObjectsEx	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
RegisterWaitForSingleObject	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
SignalObjectAndWait	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
UnregisterWait	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
UnregisterWaitEx	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
WaitForMultipleObjects	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
WaitForMultipleObjectsEx	Y	Y	Y	Y	Y	Y	Y	N	Y	N	Y	Y	Y	N	Y
WaitForSingleObject	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
WaitForSingleObjectEx	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Waitable Timers															
CancelWaitableTimer	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
CreateWaitableTimer	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
CreateWaitableTimerEx	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
OpenWaitableTimer	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
SetWaitableTimer	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Timers															
KillTimer	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
QueryPerformanceCounter	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
QueryPerformanceFrequency	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
SetTimer	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
TimerProc	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Process															

Windows API	Android	NetBSD	Linux	RTLinux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	WindowsXP/ Vista/Mobile	Windows CE	µITRON	µC/OS-III	VxWorks
CreateProcess	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ExitProcess	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
GetCommandLine	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
GetCurrentProcess	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
GetCurrentProcessId	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
GetEnvironmentStrings	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
GetPriorityClass	Y	Y	Y	Y	Y	Y	Y	N	Y	N	Y	Y	Y	N	Y
GetEnvironmentVariable	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
SetEnvironmentVariable	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
SetPriorityClass	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
TerminateProcess	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Threads															
CreateThread	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ExitThread	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
GetCurrentThread	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
GetCurrentThreadId	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
GetExitCodeThread	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
GetThreadId	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
GetThreadPriority	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
GetThreadPriorityBoost	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
ResumeThread	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
SetThreadPriority	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
SetThreadPriorityBoost	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
Sleep	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
SleepEx	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
SuspendThread	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
SwitchToThread	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
TerminateThread	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
OS_Adopt_Native_Thread_ To_Windows_Interface	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Thread Local Storage															
TlsAlloc	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
TlsFree	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
TlsGetValue	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
TlsSetValue	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Conditional Variables															

Windows API	Android	NetBSD	Linux	RTLinux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	WindowsXP/ Vista/Mobile	Windows CE	µITRON	µC/OS-III	VxWorks
AcquireSRWLockExclusive	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
AcquireSRWLockShared	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
InitializeConditionVariable	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
InitializeSRWLock	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ReleaseSRWLockExclusive	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ReleaseSRWLockShared	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
SleepConditionVariableCS	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
SleepConditionVariableSRW	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
WakeAllConditionVariable	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
WakeConditionVariable	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Error Handling															
GetLastError	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
SetLastError	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

OS Abstractor Interface

New Features Implementation —The following new features are implemented:

- **OS_Get_<Resource>_Id() API's –** In a multi process system there may be identically named resources that are owned by separate processes ,we need a way to inform the user that there are multiple resources with the same name and give them a method to find the id they are searching for. To do this, the OS_Get_<Resource>_Id() API's will need to be modified and a new set of API's that will operate on id lists need to be created.

  The API syntax will look like
  STATUS OS_Get_Pipe_Id(CHAR* User_Prefix,CHAR* name,OS_PIPE* pipe,UNSIGNED* Match_ID_Items, BOOL flag)

  The API will still search for the first id of a resource with a matching name, but the criteria for a match while in process mode will be more robust.

  For API's that do not have a scope, a check will need to be made to ensure that the resource belongs to the same process as the caller. If it does not, then it will not be considered a match. API's with this requirement are:
    - OS_Get_Dynamic_Pool_Id()
    - OS_Get_Partition_Id()
    - OS_Get_Task_Id()
    - OS_Get_Process_Id()

  If the resource has the concept of a scope, the check will be a little different. If the scope is OS_SCOPE_PROCESS, then the resource must belong to the same process as the caller to be a match. If the scope is OS_SCOPE_SYSTEM, then there is no scope matching requirement. API's with this requirement are:
    - OS_Get_Event_Group_Id()
    - OS_Get_Mutex_Id()
    - OS_Get_Pipe_Id()
    - OS_Get_Queue_Id()
    - OS_Get_Semaphore_Id()
    - OS_Get_Timer_Id()
    - OS_Get_Tiered_Memory_Pool_Id()
    - OS_Get_Tiered_Shared_Memory_Pool_Id()

  The new parameter Match_ID_Items will be an address to an UNSIGNED that will store the total number of matches. If the count parameter is null, then no count will be returned.
  The prefix & name parameters contain a path & name. If there is a path, only objects that meet that pathing criteria should be returned or considered a match for the count. The supplied path may also be partial. It will be best to work backwards from the resource name to determine the matches.

For example, lets say there is a system that has four applications that are redundant servers. Each server is identical, except for their prefix and each app is named "server". The prefixes for these apps are:

/main/comm
/reserve/comm
/backup1/out
/backup2/out

These applications all create a shared queue called "serverQ"
If the API is called as OS_Get_Queue_Id("serverQ", &id,& count); then all four queues should be considered a match and the count will contain 4.
If the API is called as OS_Get_Queue_Id("/comm/serverQ",&id,&count) then only 2 of the queue names would match, "/main/comm" and "/reserve/comm" and the count would contain 2.
If the API is called as OS_Get_Queue_Id("/main/comm/serverQ",&id,&count); then only the queue "/main/comm/serverQ" and the count would be 1
The id that is returned in the id parameter will be that of the first matching resource. If the id parameter is null, then no id is returned.

- **Prefix/Path Separator –** User can configure/set prefix path separator by configuring following macro in cross_os_usr.h header file under folder structure of "cross_os_xxx\include".

  This is applicable for Resource_ID API's, OS_Create_Process() and OS_Application_Init() to set prefix path.

```
/* seperator used in Cross OS resource prefixes */
#define OS_PREFIX_SEPERATOR          '/'
```

- **Partitions Auto Grow Feature** – Previous version of OS Abstractor implementation of partition pools (fixed size memory blocks) does not provide an option to auto-grow when it runs out of free blocks thereby making the application to stop functioning. With auto-grow feature, applications can continue to run and also will be able to correctly determine application's partition memory needs.

  The auto grow feature will have a new define in the cross_os_usr.h file to determine how to grow the pool in situations when the auto grow flag is set, however the auto grow percentage value is not provided the default value set in the usr.h (or via app_info_structure) will be used. This define will need to have the ability to be overridden by the OS_APP_INFO structure. :

  OS_PARTITION_AUTO_GROW_DELTA: This value will be used to determine how large to grow the pool. It will be a percentage of the original pool size or number of blocks to grow, whichever is higher. By default, this value will be 10 (i.e 10% of the max number of blocks specified during creation or blocks of 10 partition, whichever is higher).

- **Tiered Memory Pools –** A Tiered Memory Pool is a collection of Partition Pools of different sizes. When the user requests a segment of memory, a partition is pulled

from the best matching pool based on the block size that was requested. If no match found, then it will return an error.

- **Tiered Shared Memory Pools –** Providing a memory pool in shared memory has a major issue, especially in operating systems that use virtual memory. The problem is that when the pool is mapped into another process, there is no guarantee that the memory addresses will match. This means that we can't use the linked list nodes directly like we do for the local pools.

  Instead the partitions will be accessed array style. When a partition is allocated, a combination of the array index, pool id and the tier the partition came from will be returned as an id.

- **Adoptive Native Thread –** This function call converts a native thread, created using the APIs provided by the native OS, to a Cross OS task so the thread can use any of the API interface functions provided by Cross OS). This feature is different from the automatic adoption of the main() thread that occurs when applications call OS_Application_Init. Adopting a native thread is useful especially when you need to integrate native applications and/or libraries with OS Abstractor applications.

  Here is an example of a typical use case for this API. A native application is used to create a service provided by a dynamic link library built with Cross OS. The native application attempts to create a thread using a function provided by the DLL as an entry function. Being a native application and having no knowledge of Cross OS, the call it will use to create this thread will obviously, be a native one. This means that the newly created thread will not have a Cross OS control block and will not be able to make any Cross OS API calls. If the entry function calls this API, a Cross OS control block will be configured for it and it will then be able to call Cross OS APIs.

New APIs—The following new API has been added:

- **OS_Get_Dynamic_Pool_Id_List() –** Function to get Dynamic Pool Id List created with same resource name.
- **OS_Get_Event_Group_Id_List()-**Function to get Event Group Id List created with same resource name.
- **OS_Get_Mutex_Id_List()-**Function to get Mutex Id List created with same resource name.
- **OS_Get_Partition_Id_List()-**Function to get PartitionMemory Id List created with same resource name.
- **OS_Get_Pipe_Id_List()-**Function to get Pipe Id List created with same resource name.
- **OS_Get_Process_Id_List()-**Function to get Process Id List created with same resource name.
- **OS_Get_Queue_Id_List()-**Function to get Queue Id List created with same resource name.

- **OS_Get_Semaphore_Id_List()**-Function to get Semaphore Id List created with same resource name.
- **OS_Get_Task_Id_List()**-Function to get Task Id List created with same resource name.
- **OS_Get_Timer_Id_List()**-Function to get Timer Id List created with same resource name.
- **OS_Get_Tiered_Memory_Pool_Id_List()**-Function to get Tiered Memory Pool Id List created with same resource name.
- **OS_Get_Tiered_Shared_Memory_Pool_Id_List()**-Function to get Shared Memory Pool Id List created with same resource name.

- **OS_Get_Tiered_Memory_Pool_Id()**-Function to get Tiered Memory Pool Id & matched Items count if resource created with same resource name.
- **OS_Get_Tiered_Shared_Memory_Pool_Id() -** Function to get Tiered Shared Memory Pool Id & matched Items count if resource created with same resource name.

- **OS_Create_Tiered_Memory_Pool()**-Function to create Tiered Memory Pool.
- **OS_Create_Tiered_Shared_Memory_Pool()**-Function to create Tiered Shared Memory Pool.
- **OS_Delete_Tiered_Memory_Pool()**-Function to delete Tiered Memory Pool.
- **OS_Delete_Tiered_Shared_Memory_Pool()**-Function to delete Tiered Shared Memory Pool.
- **OS_Open_Tiered_Shared_Memory_Pool()**-Function to open Tiered Shared Memory Pool which was created previously.
- **OS_Release_Tiered_Shared_Partition()**-Function to release Tiered Shared Partition.
- **OS_Release_Tiered_Shared_Partition_Id**()-This API behaves the same as OS_Release_Tiered_Shared_Partition except that instead of taking a partition pointer as a parameter, it takes the partition id.
- **OS_Allocate_Tiered_Memory(**) - This API will allocate a partition from the tiers.
- **OS_Allocate_Tiered_Shared_Partition() –** This API will acquire a shared memory partition
- **OS_Aquire_Tiered_Shared_Partition(**) - This API will get the local address of an already allocated partition.
- **OS_Close_Tiered_Shared_Memory_Pool() -** Function to closes Tiered Shared memory pool.

- **OS_Set_Event_Group_Scope_To_System() -** Function to set the Event Group Scope from Process to System Scope.
- **OS_Set_Mutex_Scope_To_System()-** Function to set the Mutex Scope from Process to System Scope.
- **OS_Set_Pipe_Scope_To_System()-** Function to set the Pipe Scope from Process to System Scope.

- **OS_Set_Queue_Scope_To_System()-** Function to set the Queue Scope from Process to System Scope.
- **OS_Set_Semaphore_Scope_To_System()-** Function to set the Semaphore Scope from Process to System Scope.

- **OS_Adopt_Native_Thread_To_Cross_Os() –** This function call converts a native thread, created using the APIs provided by the native OS, to a Cross OS task so the thread can use any of the API interface functions provided by Cross OS.

Behavioral Changes of APIs — The following APIs behavioral changes have been made:

- **OS_Application_Init() –** This API needs to be modified to include a new parameter called app_prefix_name. The API will need to check to ensure that the combination of the prefix and the application name is unique. The prefix can be anything the end user wants. The app_prefix_name will support multiple depths based on a "/" character. An example of app_prefix_name containing three levels could be: /level1/level2/level3.

  If no prefix is specified (i.e. the passed in parameter is null) the Application will be at the root level but the name uniqueness will still apply. That is, all root level application names must be unique when compared to other root level application names.

  This change will only apply to applications built with the INCLUDE_OS_PROCESS flag set to OS_TRUE. If the flag is set to OS_FALSE no prefix is used and no checking is needed.

- **OS_Create_Process() –** This function will create a process (or a protected main task, in case the underlying OS does not support process mode). The entry function for the process will be main(). The process name (a.k.a. application name) should be unique. No resources from the current process will be copied to the new process. If you want to run applications with similar names, then you need to set the INCLUDE_OS_PROCESS configuration flag to OS_FALSE and build the application independently and pass in the application name via OS_Application_Initialize() function.

  Alternative way to create a new process would be to independently compile/link another application with OS_PROCESS feature set to OS_TRUE and call OS_Application_Initialize() function. When application initializes the OS Abstractor library, it will automatically get registered as another OS Abstractor process. The first application with OS_PROCESS set to OS_TRUE will create all the shared resources with either using the default values set in cross_os_usr.h or via the os_app_Init_Info parameter passed along to OS_Application_Init. When subsequent applications that are complied with OS_PROCESS set to OS_TRUE are loaded, they will skip the creation of the shared resources, but will automatically get registered as an OS Abstractor process.

- **OS_Create_Partition_Pool() –** Added new feature like Auto_grow flag to optionally allow the partition pool to grow.
- **OS_Allocate_Partition() –** Added new parameters.

- **OS_Get_Dynamic_Pool_Id() –** Added new parameters like matched items count & prefix path.
- **OS_Get_Event_Group_Id()–** Added new parameters like matched items count & prefix path.
- **OS_Get_Mutex_Id()–** Added new parameters like matched items count & prefix path.
- **OS_Get_Partition_Id()–** Added new parameters like matched items count & prefix path.
- **OS_Get_Pipe_Id()–** Added new parameters like matched items count & prefix path.
- **OS_Get_Process_Id()–** Added new parameters like matched items count & prefix path.
- **OS_Get_Queue_Id()–** Added new parameters like matched items count & prefix path.
- **OS_Get_Semaphore_Id()–** Added new parameters like matched items count & prefix path.
- **OS_Get_Task_Id()–** Added new parameters like matched items count & prefix path.
- **OS_Get_Timer_Id()–** Added new parameters like matched items count & prefix path.

Removed APIs—The following APIs have been deleted:

- None

Renamed APIs —The following APIs have been renamed:

- None

OS Abstractor InterfaceAPIs

The following table provides more information on OS Abstractor InterfaceAPI level of support across different target OSs.

Table 7: OS Abstractor InterfaceAPIs

Cross-OS API	Android	NetBSD	Linux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	Windows	µITRON	µC/OS-III	VxWorks
Initialization													
os_application_init.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_application_free.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_application_wait_for_end.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Process													
os_create_process.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y[10]	Y
os_get_current_process_id.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y[10]	Y
os_process_register_exit_hook.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y[10]	Y
os_delete_process.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Task													
os_create_task.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_terminate_task.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_delete_task.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_resume_task.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_set_task_priority.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_get_task_priority.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_get_current_task_id.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_relinquish_task.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_set_task_preemption.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_get_task_preemption.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_sleep_task.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_create_protection.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y[10]	Y
os_aquire_protection.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y[10]	Y
os_release_protection.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y[10]	Y
Adaptive Native Thread													
os_adopt_native_thread_to_cross_os.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Task Pool Management													

Cross-OS API	Android	NetBSD	Linux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	Windows	µITRON	µC/OS-III	VxWorks
os_add_to_task_pool.c	Y	Y	Y	Y	Y	Y	Y	N[7]	N[7]	Y	Y	Y	Y
os_remove_from_task_pool.c	Y	Y	Y	Y	Y	Y	Y	N[7]	N[7]	Y	Y	Y	Y
POSIX Interface for all Target OS Environments													
os_task_specific_error_get.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_task_specific_error_set.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_fatal_error.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Dynamic Memory Pool													
os_create_memory_pool.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_delete_memory_pool.c	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Y	Y
os_allocate_memory.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_deallocate_memory.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Partition Memory Pool													
os_create_partition_pool.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_allocate_partition.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_deallocate_partition.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_delete_partition_pool.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Tiered Memory Pool													
os_create_tiered_memory_pool.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_create_tiered_shared_memory_pool.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_delete_tiered_memory_pool.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_delete_tiered_shared_memory_pool.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_open_tiered_shared_memory_pool.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_release_tiered_shared_partition.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_release_tiered_shared_partition_id.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_allocate_tiered_memory.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_allocate_tiered_shared_partition.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_acquire_tiered_shared_partition.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_close_tiered_shared_memory_pool.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Application Timer													

Cross-OS API	Android	NetBSD	Linux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	Windows	µITRON	µC/OS-III	VxWorks
os_create_timer.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_delete_timer.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_control_timer.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_get_remaining_time.c	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Y	Y
Event													
os_create_event_group.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_delete_event_group.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_get_events.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_set_events.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Semaphores													
os_create_semaphore.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_delete_semaphore.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_get_semaphore_count.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y[9]
os_give_semaphore.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_take_semaphore.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Mutex													
os_create_mutex.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_delete_mutex.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_give_mutex.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_force_give_mutex.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_take_mutex.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_set_mutex_ceiling.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_get_mutex_ceiling.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Queues													
os_create_queue.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_delete_queue.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_send_to_queue.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_receive_from_queue.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_send_urgent_to_queue.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	Y
os_get_queue_message_count.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y
Pipes													
os_create_pipe.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Cross-OS API	Android	NetBSD	Linux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	Windows	µITRON	µC/OS-III	VxWorks
os_delete_pipe.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_send_to_pipe.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_receive_from_pipe.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_send_urgent_to_pipe.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	Y
os_get_pipe_message_count.c	N	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	Y
Clocks													
os_set_clock_ticks.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y[5]
os_get_clock_ticks.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_get_hr_clock_freq.c	N[6]	Y[4]	Y[4]	Y[4]	Y	Y	N[6]	N[6]	N[6]	Y	N[6]	N[6]	Y
os_get_calendar_time.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_set_calendar_time.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Device Drivers													
os_driver_install.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_driver_remove.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_device_add.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_device_delete.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_device_find.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_fd_value.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_default_path_get.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_default_path_set.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_create_driver_task.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_activate_driver_task.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Resource Identification													
os_get_partition_id.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_get_queue_id.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_get_dynamic_pool_id.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_get_semaphore_id.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_get_task_id.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_get_event_group_id.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_get_pipe_id.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_get_mutex_id.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_get_timer_id.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_get_process_id.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y[10]	Y

Cross-OS API	Android	NetBSD	Linux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	Windows	µITRON	µC/OS-III	VxWorks
os_get_tiered_memory_pool_id.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_get_tiered_shared_memory_pool_id.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_get_dynamic_pool_id_list.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_get_event_group_id_list.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_get_mutex_id_list.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_get_partition_id_list.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_get_pipe_id_list.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_get_process_id_list.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_get_queue_id_list.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_get_semaphore_id_list.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_get_task_id_list.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_get_timer_id_list.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_get_tiered_memory_pool_id_list.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_get_tiered_shared_memory_pool_id_list.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Link List													
os_add_to_list.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_add_to_list_by_index.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_initialize_list.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_remove_from_list.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ANSI													
os_creat.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_unlink.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_write.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_open.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_getcwd.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_getwd.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_ioctl.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_remove.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_close.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_read.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_chdir.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Cross-OS API	Android	NetBSD	Linux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	Windows	µITRON	µC/OS-III	VxWorks
ANSI Format I/O													
os_sprintf.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ANSI Memory													
os_calloc.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y[10]	Y
os_malloc.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_free.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Signal													
os_send_process_signal.c	Y	Y	Y	Y	Y	Y	N[8]	Y	N[8]	Y	Y	N[8]	Y
os_register_signal.c	Y	Y	Y	Y	Y	Y	N[8]	Y	N[8]	Y	Y	N[8]	Y
os_send_task_signal.c	Y	Y	Y	Y	Y	Y	N[8]	Y	N[8]	Y	Y	N[8]	Y
os_control_signal.c	Y	Y	Y	Y	Y	Y	N[8]	Y	N[8]	Y	Y	N[8]	Y
os_get_signal_handler.c	Y	Y	Y	Y	Y	Y	N[8]	Y	N[8]	Y	Y	N[8]	Y
Scope Change													
os_set_event_group_scope_to_system.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_set_mutex_scope_to_system.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_set_pipe_scope_to_system.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_set_queue_scope_to_system.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_set_semaphore_scope_to_system.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Serial Device													
os_setup_serial_port.c	N[2]	N[2]	N[2]	N[2]	N[2]	N[2]	N[2]	N[2]	N[2]	N[2]	N[2]	N[2]	N[2]
Miscellaneous													
os_release_information.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_get_system_info.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_create_application_counter.c	N[6]	N[1]	N[1]	N[1]	N[1]	N[1]	N[6]	N[6]	N[6]	N[1]	N[6]	N[6]	N[1]
os_decrement_application_counter.c	N[6]	N[1]	N[1]	N[1]	N[1]	N[1]	N[6]	N[6]	N[6]	N[1]	N[6]	N[6]	N[1]
os_delete_application_counter.c	N[6]	N[1]	N[1]	N[1]	N[1]	N[1]	N[6]	N[6]	N[6]	N[1]	N[6]	N[6]	N[1]
chkandgetfiledesc.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_suspend_task.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Cross-OS API	Android	NetBSD	Linux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	Windows	µITRON	µC/OS-III	VxWorks
os_error_code_string.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_get_environment.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y[10]	Y
os_increment_application_counter.c	N[6]	N[1]	N[1]	N[1]	N[1]	N[1]	N[6]	N[6]	N[6]	N[1]	N[6]	N[6]	N[1]
os_init_io.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_printf.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_pthread_kill_external.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y[10]	Y
os_put_environment.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y[10]	Y
os_read_hr_clock.c	N[6]	Y[4]	Y[4]	Y[4]	Y	Y	N[6]	N[6]	N[6]	Y	N[6]	N[6]	Y[11]
os_process_unregister_exit_hook.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y[10]	Y
os_reset_application_counter.c	N[6]	N[1]	N[1]	N[1]	N[1]	N[1]	N[6]	N[6]	N[6]	N[1]	N[6]	N[6]	N[1]
os_send_character_to_device.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_task_specific_error_get.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_task_specific_error_set.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_time_t2tm.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_tm2time_t.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_write_char_to_serial.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
OS_Monitor_Register	N	N	N	N	Y	N	N	N	N	N	N	N	N

Notes

1: stub - for future development

2: stub - This API is intended to be user configured

3: stub function

4: Only supported on x86 platforms

5: Only implemented on VxWorks 5.5 and VxWorks 6.x with kernel mode on

6: Profiler not supported on this Operating System

7: Task Pooling not supported on this Operating System

8: Signaling not supported on this Operating System

9: Only supported on VxWorks 6.x and later

10. Added in 1.3.6.1

11: Not supported in RTP mode

VxWorks Interface

New APIs—The following new APIs have been added:

- **OS_Adopt_Native_Thread_To_Vxworks_Interface() –** This API configures an adopted thread so it can use the VxWorks Interface APIs. The native thread must have been adopted by calling OS_Adopt_Native_Thread_To_Cross_OS() API prior to making this function call.

Behavioral Changes of APIs — The following APIs behavioral changes have been made:

- All Resource Create API's [Task,Sema,Mutex,Queue, Pipe…etc] are created as local [OS_SCOPE_PROCESS] by default instead of shared [OS_SCOPE_SYSTEM].

  If a customer wishes to use the added inter-process communication functionality that Cross OS provides, all they would have to do is add the appropriate API call into their code. If they do not want this, they don't have to do anything.

  Here are the list of Scope change API's:
  - OS_Set_Queue_Scope_To_System(OS_QUEUE id)
  - OS_Set_Pipe_Scope_To_System(OS_PIPE id)
  - OS_Set_Mutex_Scope_To_System(OS_MUTEX id)
  - OS_Set_Event_Group_Scope_To_System(OS_EVENT_GROUP id)
  - OS_Set_Semaphore_Scope_To_System(OS_SEMAPHORE id)

- **Mutex Priority Inversion**: Added support for SEM_INVERSION_SAFE flag in the semMCreate() function.

Removed APIs—The following APIs have been deleted:

- None

Renamed APIs —The following APIs have been renamed:

- None

VxWorks Interface APIs

The following table provides more information on VxWorks Interface API level of support across different target OSs.

Table 8: VxWorks Interface APIs

VxWorks API	Android	NetBSD	Linux	RT Linux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	Windows/Vista/XP/Mobile	Windows CE	µITRON	µC/OS-III	VxWorks
Error Handling															
errno.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N/A
errnoget.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
errnooftaskget.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
errnooftaskset.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
errnoset.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
logmsg()	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
Kernel Initialization															
kernelinit.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
kernelversion.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
Partition Memory															
memaddtopool.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N/A
mempartaddtopool.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N/A
mempartalignedalloc.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
mempartalloc.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
mempartcreate.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
mempartfree.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
Events															
eventReceive	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
eventSend	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
Queues															
msgqcreate.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
msgqdelete.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
msgqinfoget.c	N	Y[1]	N	N	N	N	Y[1]	Y[1]	Y[2]	Y[2]	N	N	Y[1]	Y[1]	N/A
msgqnummsgs.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
msgqreceive.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
msgqsend.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
msgqshow.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N/A
msgqshowinit.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N/A

VxWorks API	Android	NetBSD	Linux	RT Linux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	Windows/Vista/XP/Mobile	Windows CE	µITRON	µC/OS-III	VxWorks
Semaphores															
sembcreate.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
semccreate.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
semclear.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
semcreate.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
semdelete.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
semflush.c	N	Y	Y	Y	N	N	Y	N	N	N	N	N	N	N	N/A
semgive.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
seminfo.c	N	Y[3]	N	N	Y[3]	N	Y[3]	Y[3]	Y[3]	Y[3]	N	N	Y[3]	Y[3]	N/A
seminit.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
semmcreate.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
semmgiveforce.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
semshow.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N/A
semshowinit.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N/A
semtake.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
sysauxclkconnect.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
sysauxclkdisable.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
sysauxclkenable.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
sysauxclkrateget.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
sysauxclkrateset.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
sysclkrateget.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
Task Control Block															
taskactivate.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
taskcreatehookadd.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
exit.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
taskdelay.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
taskdelete.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
taskdeleteforce.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
taskdeletehookadd.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
taskiddefault.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N/A
taskidlistget.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
taskidself.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
taskidverify.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
taskinfoget.c	N	N	N	N	N	N	N	Y[4]	Y[4]	Y[4]	N	N	N	Y[4]	N/A
taskinit.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
taskisready.c	N	N	N	N	N	N	N	N	Y	Y	N	N	Y	N	N/A
taskissuspended.c	N	N	N	N	N	N	N	N	Y	Y	N	N	Y	N	N/A

VxWorks API	Android	NetBSD	Linux	RT Linux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	Windows/Vista/XP/Mobile	Windows CE	µITRON	µC/OS-III	VxWorks
tasklibint.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N/A
tasklock.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
taskname.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
tasknametoid.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
taskoptionsget.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
taskoptionsset.c	Y[5]	Y[5]	Y[5]	Y[5]	Y[5]	Y[5]	Y[5]	Y[5]	Y[5]	Y[5]	Y[5]	Y[5]	Y[5]	Y[5]	N/A
taskpriorityget.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
taskpriorityset.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
taskregsget.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N/A
taskregsset.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N/A
taskregsshow.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N/A
taskrestart.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
taskresume.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
tasksafe.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
taskshow.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N/A
taskshowinit.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N/A
taskspawn.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
tasksuspend.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
taskswitchhookadd.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N/A
tasktcb.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
taskunlock.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
taskunsafe.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
taskvaradd.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
taskvardelete.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
taskvarinfo.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
taskvarinit.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
taskvarset.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
Timer															
tickannounce.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N/A
tickget.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
tickset.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
IOS Library															
iosInit	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
iosDrvInstall	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
iosDrvRemove	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
iosDevAdd	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A

VxWorks API	Android	NetBSD	Linux	RT Linux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	Windows/Vista/XP/Mobile	Windows CE	µITRON	µC/OS-III	VxWorks
iosDevDelete	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
iosDevFind	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
iosEdValue	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
Watchdog Timer															
wdcancel.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
wdcreate.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
wddelete.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
wdlibinit.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N/A
wdstart.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
x_isintaskcontext.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
x_istasknotvalid.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
Adaptive Native Thread															
os_adopt_native_thread_to_vxworks_interface.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A
Notes															
1: The ifdef does not include these OS's but this API calls INT_OS_Pipe_Info, which is either unimplemented or does not return all information on these OS's															
2: Does not return maxMsgLength, options, recvTimeouts, sendTimeouts, taskIdList, msgLenList or msgPtrList															
3: Only returns the number of tasks waiting for the semaphore, none of the parameters are set															
4: Does not report td_stackMargin, td_delay, td_stackCurrent															
5: VX_UNBREAKABLE not supported															

POSIX Interface

New APIs—The following new APIs have been added:

- **OS_Adopt_Native_Thread_To_Posix_Interface() –** This API configures an adopted thread so it can use the Posix Interface APIs. The native thread must have been adopted by calling OS_Adopt_Native_Thread_To_Cross_OS() API prior to making this function call.

Behavioral Changes of APIs — The following APIs behavioral changes have been made:

- All Resource Create API's [Task,Sema,Mutex,Queue, Pipe…etc] are created as local [OS_SCOPE_PROCESS] by default instead of shared [OS_SCOPE_SYSTEM].

  If a customer wishes to use the added inter-process communication functionality that Cross OS provides, all they would have to do is add the appropriate API call into their code. If they do not want this, they don't have to do anything.

  Here are the list of Scope change API's:
  - OS_Set_Queue_Scope_To_System(OS_QUEUE id)
  - OS_Set_Pipe_Scope_To_System(OS_PIPE id)
  - OS_Set_Mutex_Scope_To_System(OS_MUTEX id)
  - OS_Set_Event_Group_Scope_To_System(OS_EVENT_GROUP id)
  - OS_Set_Semaphore_Scope_To_System(OS_SEMAPHORE id)

Removed APIs—The following APIs have been deleted:

- None

Renamed APIs —The following APIs have been renamed:

- None

POSIX Interface APIs

The following table provides more information on POSIX Interface API level of support across different target OSs.

Table 9: POSIX Interface APIs

POSIX API	Android	NetBSD	Linux	RTLinux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	WindowsXP/ Vista/Mobile	Windows CE	µITRON	µC/OS-III	VxWorks
abort.c	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	N	Y[1]	N	Y[1]	Y[1]	Y[1]	N	Y[1]
alarm.c	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	N	Y[1]	N	Y[1]	Y[1]	Y[1]	N	Y[1]
atexit.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
Clock															
clock.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
clock_getcpuclockid.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
clock_getres.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
clock_gettime.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
clock_nanosleep.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
clock_settime.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
confstr.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
continue_signal.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
dlclose.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
dlerror.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
dlopen.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
dlsym.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
errno.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
execl.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
execle.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
execlp.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
execv.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
execve.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
execvp.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
exit.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
fgets.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
fgets_remap.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
fopen.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
fopen_remap.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
fork.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
getenv.c	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]

POSIX API	Android	NetBSD	Linux	RTLinux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	WindowsXP/ Vista/Mobile	Windows CE	µITRON	µC/OS-III	VxWorks
getline.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
getpgrp.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
getpid.c	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]
getppid.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
gettimeofday.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
glob.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
globfree.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
kill.c	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	N	Y[1]	N	Y[1]	Y[1]	Y[1]	N	Y[1]
mlock.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
mlockall.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
mmap.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
mprotect.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
Queues															
mq_close.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
mq_getattr.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
mq_notify.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
mq_open.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
mq_receive.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
mq_send.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
mq_setattr.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
mq_timedreceive.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
mq_timedsend.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
mq_unlink.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
msync.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
munlock.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
munlockall.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
munmap.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
nanosleep.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_adopt_native_posix.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_posix_init.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_posix_thread_type.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
os_signal_handler_init.c	Y	Y	Y	Y	Y	Y	Y	N	Y	N	Y	Y	Y	N	Y
pause.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pipe.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
posix_spawn.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]

POSIX API	Android	NetBSD	Linux	RTLinux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	WindowsXP/ Vista/Mobile	Windows CE	µITRON	µC/OS-III	VxWorks
posix_spawnattr_destroy.c	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]
posix_spawnattr_getflags.c	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]
posix_spawnattr_getpgroup.c	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]
posix_spawnattr_getschedparam.c	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]
posix_spawnattr_getschedpolicy.c	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]
posix_spawnattr_getsigdefault.c	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]
posix_spawnattr_getsigmask.c	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]
posix_spawnattr_init.c	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]
posix_spawnattr_setflags.c	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]
posix_spawnattr_setpgroup.c	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]
posix_spawnattr_setschedparam.c	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]
posix_spawnattr_setschedpolicy.c	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]
posix_spawnattr_setsigdefault.c	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]
posix_spawnattr_setsigmask.c	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]
posix_spawnp.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
posix_spawn_file_actions_addclose.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
posix_spawn_file_actions_addopen.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
posix_spawn_file_actions_addup2.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
posix_spawn_file_actions_destroy.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
posix_spawn_file_actions_init.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
pthread_atfork.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	N	N	N	N	N	N	N	N
POSIX Threads															
pthread_create.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_attr_destroy.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_equal.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_exit.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_getconcurrency.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_getcpuclockid.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_getschedparam.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

POSIX API	Android	NetBSD	Linux	RTLinux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	WindowsXP/ Vista/Mobile	Windows CE	µITRON	µC/OS-III	VxWorks
pthread_getspecific.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_join.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_key_create.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_key_delete.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_kill.c	Y	Y	Y	Y	Y	Y	Y	N	Y	N	Y	Y	Y	N	Y
pthread_attr_getdetachstate.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_attr_getguardsize.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_attr_getinheritsched.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_attr_getschedparam.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_attr_getschedpolicy.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_attr_getscope.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_attr_getstack.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_attr_getstackaddr.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_attr_getstacksize.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_attr_init.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_attr_setdetachstate.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_attr_setguardsize.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_attr_setinheritsched.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_attr_setschedparam.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_attr_setschedpolicy.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_attr_setscope.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_attr_setstack.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_attr_setstackaddr.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_attr_setstacksize.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_self.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_setcancelstate.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_setcanceltype.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_setconcurrency.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_setschedparam.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_setschedprio.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_setspecific.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_sigmask.c	Y	Y	Y	Y	Y	Y	Y	N	Y	N	Y	Y	Y	N	Y
pthread_testcancel.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

POSIX API	Android	NetBSD	Linux	RTLinux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	WindowsXP/ Vista/Mobile	Windows CE	µITRON	µC/OS-III	VxWorks
Barriers															
pthread_barrierattr_destroy.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_barrierattr_getpshared.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_barrierattr_init.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_barrierattr_setpshared.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_barrier_destroy.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_barrier_init.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_barrier_wait.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_cancel.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_cleanup_pop.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_cleanup_push.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Conditional Variables															
pthread_condattr_destroy.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_condattr_getclock.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_condattr_getpshared.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_condattr_init.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_condattr_setclock.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_condattr_setpshared.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_cond_broadcast.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_cond_destroy.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_cond_init.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_cond_signal.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_cond_timedwait.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_cond_wait.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Mutex															
pthread_mutexattr_destroy.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_mutexattr_getpriorceiling.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_mutexattr_getprotocol.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_mutexattr_getshare.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_mutexattr_gettype	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

POSIX API	Android	NetBSD	Linux	RTLinux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	WindowsXP/ Vista/Mobile	Windows CE	µITRON	µC/OS-III	VxWorks
.c															
pthread_mutexattr_init.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_mutexattr_setprioceiling.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_mutexattr_setprotocol.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_mutexattr_setpshared.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_mutexattr_settype.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_mutex_destroy.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_mutex_getprioceiling.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_mutex_init.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_mutex_lock.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_mutex_setprioceiling.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_mutex_timedlock.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_mutex_trylock.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_mutex_unlock.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_once.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
R/W Locks															
pthread_rwlockattr_destroy.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_rwlockattr_getpshared.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_rwlockattr_init.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_rwlockattr_setpshared.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_rwlock_destroy.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_rwlock_init.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_rwlock_rdlock.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_rwlock_timedrdlock.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_rwlock_timedwrlock.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_rwlock_tryrdlock.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_rwlock_trywrlock.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_rwlock_unlock.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_rwlock_wrlock.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Spin-Locks															

POSIX API	Android	NetBSD	Linux	RTLinux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	WindowsXP/ Vista/Mobile	Windows CE	µITRON	µC/OS-III	VxWorks
pthread_spin_destroy.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_spin_init.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_spin_lock.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_spin_trylock.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pthread_spin_unlock.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
putenv.c	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]
raise.c	Y	Y	Y	Y	Y	Y	Y	N	Y	N	Y	Y	Y	N	Y
regcomp.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
regerror.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
regexec.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
regfree.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
sched_getparam.c	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]
sched_getscheduler.c	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]
sched_get_priority_max.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
sched_get_priority_min.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
sched_rr_get_interval.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
sched_setparam.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
sched_setscheduler.c	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]
sched_yield.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Semaphores															
sem_close.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
sem_destroy.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
sem_getvalue.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
sem_init.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
sem_open.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
sem_post.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
sem_timedwait.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
sem_trywait.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
sem_unlink.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
sem_wait.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
setsid.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
shm_open.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
shm_unlink.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
sigaction.c	Y	Y	Y	Y	Y	Y	Y	N	Y	N	Y	Y	Y	N	Y
sigaddset.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

POSIX API	Android	NetBSD	Linux	RTLinux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	WindowsXP/ Vista/Mobile	Windows CE	µITRON	µC/OS-III	VxWorks
sigaltstack.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
sigdelset.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
sigemptyset.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
sigfillset.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
sighold.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
sigignore.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
siginterrupt.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
sigismember.c	Y	Y	Y	Y	Y	Y	Y	N	Y	N	Y	Y	Y	N	Y
siglongjmp.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
signal.c	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	N	Y[1]	N	Y[1]	Y[1]	Y[1]	N	Y[1]
sigpause.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
sigpending.c	Y	Y	Y	Y	Y	Y	Y	N	Y	N	Y	Y	Y	N	Y
sigprocmask.c	Y	Y	Y	Y	Y	Y	Y	N	Y	N	Y	Y	Y	N	Y
sigqueue.c	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	N	Y[1]	N	Y[1]	Y[1]	Y[1]	N	Y[1]
sigrelse.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
sigsetjmp.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
sigsuspend.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
sigtimedwait.c	Y	Y	Y	Y	Y	Y	Y	N	Y	N	Y	Y	Y	N	Y
sigwait.c	Y	Y	Y	Y	Y	Y	Y	N	Y	N	Y	Y	Y	N	Y
sigwaitinfo.c	Y	Y	Y	Y	Y	Y	Y	N	Y	N	Y	Y	Y	N	Y
sleep.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
sysconf.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
Timers															
timer_create.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
timer_delete.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
timer_getoverrun.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
timer_gettime.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
timer_settime.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
times.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
Adaptive Native Thread															
os_adopt_native_thread_to_posix_interface.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Miscellaneous															
uname.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
unsetenv.c	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]	Y[1]

POSIX API	Android	NetBSD	Linux	RTLinux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	WindowsXP/ Vista/Mobile	Windows CE	µITRON	µC/OS-III	VxWorks
usleep.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
wait.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
waitpid.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
_exit.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
NOTES															
1: Only allowed when INCLUDE_OS_PROCESS is set to OS_TRUE															
2: Included if the underlying OS has this functionality. These functions are mapped directly to the underlying OS's version.															

Nucleus Interface

New APIs—The following new APIs have been added:

- **OS_Adopt_Native_Thread_To_Nucleus_Interface() –** This API configures an adopted thread so it can use the Nucleus Interface APIs. The native thread must have been adopted by calling OS_Adopt_Native_Thread_To_Cross_OS() API prior to making this function call.

Behavioral Changes of APIs — The following APIs behavioral changes have been made:

- All Resource Create API's [Task,Sema,Mutex,Queue, Pipe…etc] are created as local [OS_SCOPE_PROCESS] by default instead of shared [OS_SCOPE_SYSTEM].

  If a customer wishes to use the added inter-process communication functionality that Cross OS provides, all they would have to do is add the appropriate API call into their code. If they do not want this, they don't have to do anything.

  Here are the list of Scope change API's:
  - OS_Set_Queue_Scope_To_System(OS_QUEUE id)
  - OS_Set_Pipe_Scope_To_System(OS_PIPE id)
  - OS_Set_Mutex_Scope_To_System(OS_MUTEX id)
  - OS_Set_Event_Group_Scope_To_System(OS_EVENT_GROUP id)
  - OS_Set_Semaphore_Scope_To_System(OS_SEMAPHORE id)

Removed APIs—The following APIs have been deleted:

- None

Renamed APIs —The following APIs have been renamed:

- None

Nucleus Interface APIs

The following table provides more information on Nucleus Interface API level of support across different target OSs.

Table 10: Nucleus Interface APIs

Nucleus API	Android	NetBSD	Linux	RT Linux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	Windows XP /Mobile/Vista	µITRON	µC/OS-III	VxWorks
Tasks														
nu_change_preemption.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_change_priority.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_create_task.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_current_task_pointer.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_delete_task.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_relinquish.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_reset_task.c	N	N	N	N	N	N	N	N	N/A	N	N	N	N	N
nu_resume_task.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_sleep.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_suspend_task.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_terminate_task.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_task_information.c	N	N	Y[6]	Y[6]	N	N	N	N	N/A	N	N	N	N	N
nu_task_pointers.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_established_tasks.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
Message Queues														
nu_create_queue.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_delete_queue.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_queue_information.c	N	N	Y[4]	Y[4]	N	N	N	N	N/A	N	N	N	N	N
nu_queue_pointers.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_receive_from_queue.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_send_to_queue.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_send_to_front_of_queue.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
Message Pipes														
nu_create_pipe.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_delete_pipe.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_pipe_information.c	Y[3]	Y[3]	Y[3]	Y[3]	Y[3]	Y[3]	Y[3]	Y[3]	N/A	Y[3]	Y[3]	Y[3]	Y[3]	Y[3]
nu_pipe_pointers.c	Y[3]	Y[3]	Y[3]	Y[3]	Y[3]	Y[3]	Y[3]	Y[3]	N/A	Y[3]	Y[3]	Y[3]	Y[3]	Y[3]
nu_receive_from_pipe.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y

Nucleus API	Android	NetBSD	Linux	RT Linux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	Windows XP /Mobile/Vista	µITRON	µC/OS-III	VxWorks
nu_send_to_front_of_pipe.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_send_to_pipe.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
Semaphores														
nu_create_semaphore.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_delete_semaphore.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_obtain_semaphore.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_release_semaphore.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_semaphore_information.c	N	N	Y[5]	Y[5]	N	N	N	N	N/A	N	N	N	N	N
nu_semaphore_pointers.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
Events														
nu_create_event_group.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_delete_event_group.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_retrieve_events.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_set_events.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
Partition Memory Pools														
nu_allocate_partition.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_create_partition_pool.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_deallocate_partition.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_delete_partition_pool.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_partition_pool_information.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	N/A	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
Dynamic Memory Pools														
nu_create_memory_pool.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_allocate_memory.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_delete_memory_pool.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_deallocate_memory.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_memory_pool_information.c	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]	N/A	Y[2]	Y[2]	Y[2]	Y[2]	Y[2]
Timers														
nu_control_timer.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_create_timer.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_delete_timer.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_reset_timer.c	N	N	Y	Y	N	N	N	N	N/A	N	N	N	N	N

Nucleus API	Android	NetBSD	Linux	RT Linux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	Windows XP /Mobile/Vista	µITRON	µC/OS-III	VxWorks
nu_timer_pointers.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_set_clock.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_retrieve_clock.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_timer_information.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
Drivers														
nu_create_driver.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_delete_driver.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_request_driver.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
HISRS														
nu_activate_hisr.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_create_hisr.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_current_hisr_pointer.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_hisr_entry.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_hisr_information.c	N	N	Y[1]	Y[1]	N	N	N	N	N/A	N	N	N	N	N
nu_hisr_pointers.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
Miscellaneous														
nu_protect.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
erc_system_error.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_release_information.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
nu_sd_init_port.c	N	N	N	N	N	N	N	N	N/A	N	N	N	N	N
nu_sd_put_char.c	N	N	N	N	N	N	N	N	N/A	N	N	N	N	N
nu_sd_put_string.c	N	N	N	N	N	N	N	N	N/A	N	N	N	N	N
nu_unprotect.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
tcce_suspend_error.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y
Adopt Native Thread														
os_adopt_native_thread_to_nucleus_interface.c	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Y	Y	Y	Y	Y

Notes
1: Minimum stack not reported
2: start_address, available, allocated, tasks_waiting and first_task not reported
3: start_address, pipe_size, available, messages, message_type, suspend_type, tasks_waiting and first_task not reported
4: tasks_waiting, start_address and first_task not reported

Nucleus API	Android	NetBSD	Linux	RT Linux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	Windows XP /Mobile/Vista	µITRON	µC/OS-III	VxWorks
5: first_task not reported														
6: task_status, scheduled_count and minimum_stack not reported														

pSOS Interface

New APIs—The following new APIs have been added:

- **OS_Adopt_Native_Thread_To_Psos_Interface() –** This API configures an adopted thread so it can use the Psos Interface APIs. The native thread must have been adopted by calling OS_Adopt_Native_Thread_To_Cross_OS() API prior to making this function call.

- **de_write –** Device write I/O function which was missing earlier got added

Behavioral Changes of APIs — The following APIs behavioral changes have been made:

- All Resource Create API's [Task,Sema,Mutex,Queue, Pipe…etc] are created as local [OS_SCOPE_PROCESS] by default instead of shared [OS_SCOPE_SYSTEM].

  If a customer wishes to use the added inter-process communication functionality that Cross OS provides, all they would have to do is add the appropriate API call into their code. If they do not want this, they don't have to do anything.

  Here are the list of Scope change API's:
  - OS_Set_Queue_Scope_To_System(OS_QUEUE id)
  - OS_Set_Pipe_Scope_To_System(OS_PIPE id)
  - OS_Set_Mutex_Scope_To_System(OS_MUTEX id)
  - OS_Set_Event_Group_Scope_To_System(OS_EVENT_GROUP id)
  - OS_Set_Semaphore_Scope_To_System(OS_SEMAPHORE id)

Removed APIs—The following APIs have been deleted:

- None

Renamed APIs —The following APIs have been renamed:

- None

pSOS Interface APIs

The following table provides more information on pSOS Interface API level of support across different target Oss.

Table 11: pSOS Interface APIs

pSOS API	Android	NetBSD	Linux	RT Linux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	Windows XP/ Vista/Mobile	Windows CE	µITRON	µC/OS-III	VxWorks
Signals															
as_catch.c	N	N	N	N	N	N	N	N	Y	N	N	N	N	N	N
as_return.c	N	N	N	N	N	N	N	N	Y	N	N	N	N	N	N

pSOS API	Android	NetBSD	Linux	RT Linux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	Windows XP/ Vista/Mobile	Windows CE	µITRON	µC/OS-III	VxWorks
as_send.c	N	N	N	N	N	N	N	N	Y	N	N	N	N	N	N
Device I/O Interfaces															
de_close.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
de_cntrl.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
de_init.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
de_open.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
de_read.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
de_write	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Events															
ev_asend.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ev_receive.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ev_send.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
executeinttimer.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Initialization															
installdriver.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ps_initialize.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Partition Memory															
pt_create.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pt_delete.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pt_getbuf.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pt_ident.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pt_retbuf.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pt_sgetbuf.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Fixed Queues															
q_broadcast.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
q_create.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

pSOS API	Android	NetBSD	Linux	RT Linux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	Windows XP/ Vista/Mobile	Windows CE	µITRON	µC/OS-III	VxWorks
q_delete.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
q_ident.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
q_receive.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
q_send.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
q_urgent.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
q_vbroadcast.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
q_vcreate.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
q_vdelete.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
q_vident.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
q_vreceive.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
q_vsend.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
q_vurgent.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Region Memory															
rn_create.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
rn_delete.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
rn_getseg.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
rn_ident.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
rn_retseg.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Semaphores															
sm_create.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
sm_delete.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
sm_ident.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
sm_p.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
sm_v.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Timers															
tm_cancel.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
tm_evafter.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
tm_evevery.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
tm_evwhen.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N

pSOS API	Android	NetBSD	Linux	RT Linux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	Windows XP/ Vista/Mobile	Windows CE	µITRON	µC/OS-III	VxWorks
tm_wkafter.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
tm_wkwhen.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
Task Control															
t_create.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
t_delete.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
t_getreg.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
t_ident.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
t_mode.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
t_restart.c	N	N	N	N	N	N	N	Y	Y	Y	Y	Y	N	Y	Y
t_resume.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
t_setpri.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
t_setreg.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
t_start.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
t_suspend.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Adaptive Native Thread															
os_adopt_native_thread _to_psos_interface.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

pSOS 1.4 - pSOS Classic Interface

pSOS Classic interface component provides compliance to pSOS 1.4, date 3/10/1986 release.

New APIs—The following new APIs have been added:

- None

Removed APIs—The following APIs have been deleted:

- None

Renamed APIs —The following APIs have been renamed:

- None

pSOS 1.4 - pSOS Classic Interface APIs

The following table provides more information on pSOS Interface API level of support across different target Oss.

Table 12: pSOS 1.4 - pSOS Classic Interface APIs

pSOS 1.4 - pSOS Classic API	Android	NetBSD	Linux	RT linux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	Windows XP/ Vista/Mobile	Windows CE	µITRON	µC/OS-III	VxWorks
Task Control															
activate_p	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
delete_p	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ident_p	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
mode_p	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pause_p	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
priority_p	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
resume_p	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
spawn_p	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
super_p	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
suspend_p	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Messaging															
attach_x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
create_x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
delete_x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
jam_x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
liber_x	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
req_x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Signal handling															

pSOS 1.4 - pSOS Classic API	Android	NetBSD	Linux	RT linux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	Windows XP/ Vista/Mobile	Windows CE	µITRON	µC/OS-III	VxWorks
send_x	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
signal_v	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
get_v	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
wait_v	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Memory Management															
alloc_seg	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
assign_seg	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
grab_seg	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
free_seg	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Miscellaneous															
announce_t	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
get_t	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
set_t	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

µITRON Interface

New APIs—The following new APIs have been added:

- **OS_Adopt_Native_Thread_To_Uitron_Interface() –** This API configures an adopted thread so it can use the Uitron Interface APIs. The native thread must have been adopted by calling OS_Adopt_Native_Thread_To_Cross_OS() API prior to making this function call.

Behavioral Changes of APIs — The following APIs behavioral changes have been made:

- All Resource Create API's [Task,Sema,Mutex,Queue, Pipe…etc] are created as local [OS_SCOPE_PROCESS] by default instead of shared [OS_SCOPE_SYSTEM].

  If a customer wishes to use the added inter-process communication functionality that Cross OS provides, all they would have to do is add the appropriate API call into their code. If they do not want this, they don't have to do anything.

  Here are the list of Scope change API's:
  - OS_Set_Queue_Scope_To_System(OS_QUEUE id)
  - OS_Set_Pipe_Scope_To_System(OS_PIPE id)
  - OS_Set_Mutex_Scope_To_System(OS_MUTEX id)
  - OS_Set_Event_Group_Scope_To_System(OS_EVENT_GROUP id)
  - OS_Set_Semaphore_Scope_To_System(OS_SEMAPHORE id)

Removed APIs—The following APIs have been deleted:

- None

Renamed APIs —The following APIs have been renamed:

- None

µITRON Interface APIs

The following table provides more information on µITRON Interface API level of support across different target Oss.

Table 13: µITRON Interface APIs

µITRON API	Android	NetBSD	Linux	RT Linux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	WindowsXP/ Vista/Mobile	Windows CE	µC/OS-III	VxWorks
Task Management Functions														
acre_tsk.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y
act_tsk.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y
del_tsk.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y
cre_tsk.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y
exd_tsk.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y
ext_tsk.c	N	N	N	N	N	N	N	N	N	N	N		N	N
iact_tsk.c	N	N	N	N	N	N	N	N	N	N	N		N	N
sta_tsk.c	N	N	N	N	N	N	N	N	N	N	N		N	N
ref_tsk.c	N	N	N	N	N	N	N	N	N	N	N		N	N
chg_pri.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y
iwup_tsk.c	N	N	N	N	N	N	N	N	N	N	N		N	N
ter_tsk.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y
get_pri.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y
ref_tst.c	N	N	N	N	N	N	N	N	N	N	N		N	N
Task Dependent Synchronization Functions														
slp_tsk.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y
tslp_tsk.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y
wup_tsk.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y
iwup_tsk.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y
can_wup.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y
rel_wai.c	N	N	N	N	N	N	N	N	N	N	N		N	N
irel_wai.c	N	N	N	N	N	N	N	N	N	N	N		N	N
sus_tsk.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y
rsm_tsk.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y
frsm_tsk.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y
dly_tsk.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y
Task Exception Handling Functions														
def_tex.c	N	N	N	N	N	N	N	N	N	N	N		N	N
dis_tex.c	N	N	N	N	N	N	N	N	N	N	N		N	N

µITRON API	Android	NetBSD	Linux	RT Linux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	WindowsXP/ Vista/Mobile	Windows CE	µC/OS-III	VxWorks
ena_tex.c	N	N	N	N	N	N	N	N	N	N	N		N	N
ras_tex.c	N	N	N	N	N	N	N	N	N	N	N		N	N
ires_tex.c	N	N	N	N	N	N	N	N	N	N	N		N	N
ref_tex.c	N	N	N	N	N	N	N	N	N	N	N		N	N
sns_tex.c	N	N	N	N	N	N	N	N	N	N	N		N	N
Semaphores														
acre_sem.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y
cre_sem.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y
del_sem.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y
isig_sem.c	N	N	N	N	N	N	N	N	N	N	N		N	N
ref_sem.c	N	N	N	N	N	N	N	N	N	N	N		N	N
sig_sem.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y
pol_sem.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y
twai_sem.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y
wai_sem.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y
Event Flags														
acre_flg.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y
clr_flg.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
cre_flg.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
del_flg.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
iset_flg.c	N	N	N	N	N	N	N	N	N	N	N	Y	N	N
pol_flg.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y
ref_flg.c	N	N	N	N	N	N	N	N	N	N	N	Y	N	N
set_flg.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y
twai_flg.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
wai_flg.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
												Y		
Data Queues														
acre_dtq.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y
cre_dtq.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
del_dtq.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
fsnd_dtq.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ifsnd_dtq.c	N	N	N	N	N	N	N	N	N	N	N	Y	N	N
ipsnd_dtq.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
prcv_dtq.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y
psnd_dtq.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

µITRON API	Android	NetBSD	Linux	RT Linux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	WindowsXP/ Vista/Mobile	Windows CE	µC/OS-III	VxWorks
rcv_dtq.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ref_dtq.c	N	N	N	N	N	N	N	N	N	N	N	Y	N	N
snd_dtq.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y
trcv_dtq.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
tsnd_dtq.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
												Y		
Mailboxes														
acre_mbx.c	N	N	N	N	N	N	N	N	N	N	N		N	N
cre_mbx.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
del_mbx.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
rcv_mbx.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
ref_mbx.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
snd_mbx.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
prcv_mbx.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
trcv_mbx.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
												N		
Mutexea														
acre_mtx.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y
cre_mtx.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
del_mtx.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ploc_mtx.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
loc_mtx.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ref_mtx.c	N	N	N	N	N	N	N	N	N	N	N	Y	N	N
tloc_mtx.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y
unl_mtx.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
												Y		
Message Buffers														
acre_mbf.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y
cre_mbf.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
del_mbf.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
prcv_mbf.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ref_mbf.c	N	N	N	N	N	N	N	N	N	N	N	Y	N	N
psnd_mbf.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y
trcv_mbf.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
tsnd_mbf.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
rcv_mbf.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
snd_mbf.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
												Y		

µITRON API	Android	NetBSD	Linux	RT Linux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	WindowsXP/ Vista/Mobile	Windows CE	µC/OS-III	VxWorks
Rendezvous														
acp_por.c	N	N	N	N	N	N	N	N	N	N	N		N	N
acre_por.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
cal_por.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
cre_por.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
fwd_por.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
ref_por.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
pacp_por.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
pacp_por.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
tacp_por.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
tcal_por.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
ref_rdv.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
rpl_rdv.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
del_por.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
												N		
Fixed-Sized Memory Pool														
acre_mpf.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y
cre_mpf.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
del_mpf.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
get_mpf.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pget_mpf.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ref_mpf.c	N	N	N	N	N	N	N	N	N	N	N	Y	N	N
rel_mpf.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y
tget_mpf.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
												Y		
Variable-Sized Memory Pools														
acre_mpl.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y
cre_mpl.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
del_mpl.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
get_mpl.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
pget_mpl.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ref_mpl.c	N	N	N	N	N	N	N	N	N	N	N	Y	N	N
rel_mpl.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y
tget_mpl.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
												Y		
System Time Management														
get_tim.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y

µITRON API	Android	NetBSD	Linux	RT Linux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	WindowsXP/ Vista/Mobile	Windows CE	µC/OS-III	VxWorks
isig_tim.c	N	N	N	N	N	N	N	N	N	N	N	Y	N	N
set_tim.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y
												Y		
Cyclic Handlers														
acre_cyc.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y
cre_cyc.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
del_cyc.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ref_cyc.c	N	N	N	N	N	N	N	N	N	N	N	Y	N	N
stp_cyc.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y
sta_cyc.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
												Y		
Alarm Handlers														
acre_alm.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y
cre_alm.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
del_alm.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ref_alm.c	N	N	N	N	N	N	N	N	N	N	N	Y	N	N
sta_alm.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y
stp_alm.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
												Y		
Overrun Handlers														
def_ovr.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
ref_ovr.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
sta_ovr.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
stp_ovr.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
System State Management Functions														
irot_rdq.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
rot_rdq.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
get_tid.c	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y
iget_tid.c	N	N	N	N	N	N	N	N	N	N	N	Y	N	N
iloc_cpu.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
iunl_cpu.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
loc_cpu.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
unl_cpu.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
ena_dsp.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
dis_dsp.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
sns_dsp.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N

µITRON API	Android	NetBSD	Linux	RT Linux	LynxOS	QNX	Solaris	MQX	Nucleus	ThreadX	WindowsXP/ Vista/Mobile	Windows CE	µC/OS-III	VxWorks
sns_ctx.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
sns_loc.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
sns_dpn.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
ref_sys.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
Interrupt Management Functions														
def_inh.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
cre_isr.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
acre_isr.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
ref_isr.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
del_isr.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
dis_int.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
ena_int.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
chg_ixx.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
get_ixx.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
Service Call Management Functions														
def_svc.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
cal_svc.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
System Configuration Management Functions														
can_act.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
def_exc.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
ref_cfg.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
ref_ver.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N
Adaptive Native Thread														
os_adopt_native_thread_to_uitron_interface.c	N	N	N	N	N	N	N	N	N	N	N	N	N	N

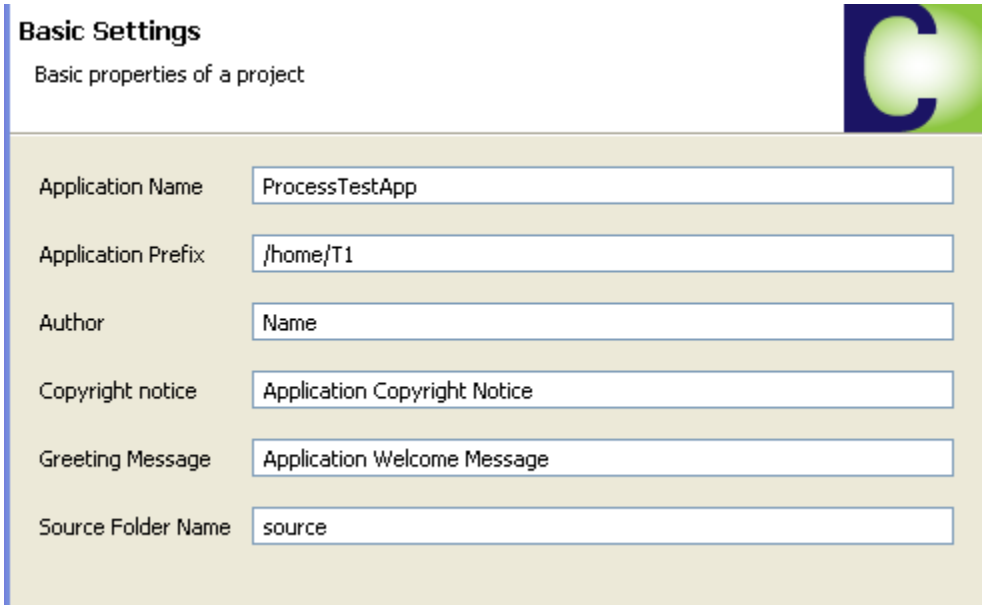
Bugs Fixed

- **Vxworks Interface related system Auxilary clock** related bug fixed**.**

APPCOE IDE

When creating any APPCOE C/C++ template project, user can set properties like Application Prefix & Name along with pre-existing properties.

Here is the screen shot of the Basic settings window.



Issues While Upgrading APPCOE from 1.3.7 to 1.3.8

The following are the issues you may face while upgrading APPCOE from Release 1.3.7 to 1.3.8:

- **APPCOE Upgradation**: You cannot use the upgrade option in APPCOE to upgrade to 1.3.8. You must take/download a fresh release of APPCOE 1.3.8.
  Note: Only patch releases can be upgraded.

Known Issues

Release 1.3.8 has the following known limitations:

- Profiler Feature is not supported in Nucleus, ThreadX, µC/OS-III and VxWorks RTP targets in this Release.
- Task pooling feature is not supported in Nucleus, ThreadX, and µC/OS-III targets in this release.
- Creating API Profiling functions while creating a C Project, we do not support overloaded functions.
- Self deletion of POSIX thread is not supported in Nucleus target.

- Application for VxWorks 6.7 should avoid defining XOPEN_SOURCE to 600.
- You cannot rebuild the canned demos as there is a soft link to Ada source. But as you need the binaries to run/debug, try to remove the soft link or try to add the actual source inside the adaRoot directory.
- "Update Settings" Option will not work for ADA projects, related Bug#962.
- win32host/Legacy Porting: You will have issue with "Import Legacy code", pulling everything include make files causing build errors, related Bug#958..
- You will be unable to run ADA project generated using Tokeneer with Abstractor for both ADA C/C++Changer and Ada-PAL Compiler on 64 bit machine on Windows 7 using Windows Build. As a workaround, to make it run on Windows 7 machine, we need to turn OFF/disable the UAC [User Account Control], related Bug#989.
- When you generate full package and build the project on Momentics IDE for QNX target with all interfaces enabled, you will get compilation errors for all interfaces except cross_os. In order to force Momentics to update these paths, right-click on the project and choose Properties from the context menu. Then click the Apply button and close the properties window, related Bug#981.
- When you do target code generation for Ada-C/C++ Changer projects along with Abstractor, it will generate sample project files. You have to generate your own project files to generate binaries other than Windows and Linux Target.
- For Ada C/C++Changer project, from Properties page if you change Ada Main procedure, it will not build the project with that procedure immediately. You need to select the project and refresh 1-2 times and clean the project and then do the build.
- API optimization is not supported for APPCOE libraries linked with application project during target code generation.
- System hangs while Target Code Optimized application runs on Linux target. As a workaround do not terminate the application till profiler is generated. If you terminate in between, your PC hangs, related Bug#1033.
- The profiler feature does not generate profiler file XXX.PAL on Solaris target if you do code optimization for demo_cross_os with profiler ON. As a workaround, enter the following command at the prompt prior to running the demo:

  prctl -n process.max-msg-qbytes -r -v 512KB -i process $$

  The 512KB is the desired size of the queue and should be sufficient to run this example. If the number of messages is increased in cross_os_usr.h, then obviously this value will need to be adjusted, related Bug#987.
- Ada-C/C++ Changer only supports Ada to 'C' and not Ada to 'C++' under Microsoft's .NET tools. Also, the GNAT compatibility feature will also not be supported under Microsoft's .NET tools. However, these features are all supported when using GNU tools for the windows platform.
- Currently external malloc() function does not supported so make “OS_USE_EXTERNAL_MALLOC” macro to OS_FALSE in cross_os_usr.h file [It is default configuration]

- Ada-C/C++ Changer projects, created with C/C++ project name more than 72 characters, will have build errors, related Bug#1321.
- It may be necessary to refresh the Windriver Workbench workspace after importing optimized RTP project files in order for project references to resolve correctly.

Release 1.3.8 Host System Requirements

No	Supported Host Platforms	System Requirements
1	Windows 8, 7,XP, Vista	Minimum 1 GB RAM
2	Linux 2.6 or Later	Minimum 1 GB RAM

To run APPCOE 1.3.8:

You can run APPCOE 1.3.8 on all the following configurations on both windows and Linux host platform:

- 64 bit hardware and 32 bit OS
- 64 bit hardware and 64 bit OS
- 32 bit hardware and 32 bit OS
- Make sure you have the necessary executable permissions to run in Linux. Make sure you have read/write privileges to the APPCOE installation directories so that the *appcoe.exe* application is able to create files and such while running
- If you experience the error [gdb: unknown target exception 0xc0000135 at 0x7c9666c6], then more likely the environmental PATH variable is set to pull the incorrect path settings for mingw/bin and msys/bin. Please ensure that the PATH is set correctly to the following:

  <INSTALDIR>mingw

  <INSTALDIR>mingw/bin

  <INSTALDIR>msys/1.0/bin

  Or try upgrading the old mingw and msys to this [mingw > gcc 4.5.2, gdb 7.4 & msys 1.0] specified version.

Technical Support

Technical support is available through the MapuSoft Technologies Support Centre. If you are a customer with an active MapuSoft support contract, or covered under warranty, and need post sales technical support, you can access our tools and resources online or open a ticket at https://www.mapusoft.com/support.

Revision History

December 2012 – Revision 1, Release 1.3.8 of MapuSoft Technologies.