RTI Connext

Core Libraries and Utilities

Platform Notes

Version 5.0

Your systems. Working as one.

Printed in U.S.A. First printing.
August 2012.

Trademarks
Real-Time Innovations, RTI, DataBus, and Connext are trademarks or registered trademarks of Real-Time Innovations, Inc. All other trademarks used in this document are the property of their respective owners.

Copy and Use Restrictions
No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form (including electronic, mechanical, photocopy, and facsimile) without the prior written permission of Real-Time Innovations, Inc. The software described in this document is furnished under and subject to the RTI software license agreement. The software may be used or copied only under the terms of the license agreement.

Technical Support
Real-Time Innovations, Inc.
232 E. Java Drive
Sunnyvale, CA 94089
Phone: (408) 990-7444
Email: support@rti.com
Website: https://support.rti.com/

Contents

1 Supported Platforms 1

2 AIX Platforms 3

2.1 Changing Thread Priority 3
2.2 Multicast Support 3
2.3 Supported Transports 3
2.4 Monotonic Clock Support 4
2.5 Support for Controlling CPU Core Affinity for RTI Threads 4

3 INTEGRITY Platforms 7

3.1 Patch Required for INTEGRITY 10.02 Platform 8
3.2 Support for Request-Reply Communication Pattern 8
3.3 Diagnostics on INTEGRITY Systems 8
3.4 Socket-Enabled and POSIX-Enabled Threads are Required 9
3.5 Running over IP Backplane on a Dy4 Champ-AVII Board 9
3.6 Multi-NIC Support on INTEGRITY 5.0 9
3.7 Multicast Support 9
3.8 Supported Transports 9
3.9 Using rtiddsping and rtiddsspy on PowerPC INTEGRITY Systems 11
3.10 Monotonic Clock Support 11
3.11 Support for Controlling CPU Core Affinity for RTI Threads 11
3.12 Issues with INTEGRITY Systems 11

4 Linux and Fedora Platforms 13

4.1 Native POSIX Thread Library (NPTL) Requirements 13
4.2 Multicast Support 14
4.3 Supported Transports 14
4.4 Monotonic Clock Support 14
4.5 Support for Controlling CPU Core Affinity for RTI Threads 14
4.6 Libraries Required for Using RTI Secure WAN Transport APIs 14
4.7 Libraries Required for Using RTI TCP Transport APIs 15

5 LynxOS Platforms 23

5.1 Multicast Support 23
5.2 Supported Transports 23
5.3 Monotonic Clock Support 24
5.4 Support for Controlling CPU Core Affinity for RTI Threads 24

6 Mac OS Platforms 27

6.1 Multicast Support 27
6.2 Supported Transports 27
6.3 Monotonic Clock Support 27
6.4 Support for Controlling CPU Core Affinity for RTI Threads 27

7 QNX Platforms....31
7.1 Required Change for Building with C++ Libraries for QNX Platforms—New in 5.0.0....31
7.2 Multicast Support....31
7.3 Supported Transports....31
7.4 Monotonic Clock Support....32
7.5 Support for Controlling CPU Core Affinity for RTI Threads....32
7.6 Restarting Applications on QNX Systems....32

8 Solaris Platforms....35
8.1 Support for Request-Reply Communication Pattern....35
8.2 Multicast Support....35
8.3 Supported Transports....35
8.4 Monotonic Clock Support....36
8.5 Support for Controlling CPU Core Affinity for RTI Threads....37
8.6 Libraries Required for using RTI Secure WAN Transport APIs....37

9 VxWorks Platforms....41
9.1 Support for Request-Reply Communication Pattern....41
9.2 Increasing the Stack Size....41
9.3 Libraries for RTP on VxWorks 6....41
9.4 Requirement for Restarting Applications....42
9.5 Multicast Support....42
9.6 Supported Transports....42
9.7 Monotonic Clock Support....42
9.8 Support for Controlling CPU Core Affinity for RTI Threads....42

10 Windows Platforms....57
10.1 Use Dynamic MFC Library, Not Static....60
10.2 Visual Studio 2005 Required when Using RTI 'Debug' Libraries for Java or .NET APIs....60
10.3 .NET API Requires Thread Affinity....60
10.4 Multicast Support....60
10.5 Supported Transports....61
10.6 Monotonic Clock Support....61
10.7 Support for Controlling CPU Core Affinity for RTI Threads....61
10.8 PPP Link Support for Windows XP Systems....61
10.9 Libraries Required for Using RTI Secure WAN Transport APIs....61
10.10 Libraries Required for Using RTI TCP Transport APIs....62

11 Custom Supported Platforms....70

Platform Notes

This document provides platform-specific instructions on how to compile, link, and run *RTI® Connext™* (formerly *RTI Data Distribution Service*) applications.

1 Supported Platforms

Table 1.1 lists the platforms available with *Connext* 5.0.

See also: Custom Supported Platforms (Section 11).

Table 1.1 **Platforms Available with Release 5.0**

Operating System		Reference
AIX®	AIX 5.3	Table 2.1 on page 4
INTEGRITY®	INTEGRITY 5.0.11 and 10.0.2	Table 3.1 on page 7
Linux® (Cell BE™)	Fedora™ 12 (2.6.32 kernel)	Table 4.1 on page 15
Linux (Intel®)	CentOS 5.4, 5.5, 6.0 Fedora 12 (2.6.32 kernel) Fedora 12 (2.6.32 kernel) with gcc 4.5.1 Red Hat® Enterprise Linux 5.0-5.2, 5.4, 5.5, 6.0, 6.1 Red Hat Enterprise Linux 5.2 with Real-Time Extensions SUSE® Linux Enterprise Server 10.1 (2.6 kernel) Ubuntu® Server 10.04 (2.6 kernel) Wind River® Linux 4 (2.6 kernel)	Table 4.2 on page 15
Linux (PowerPC®)	Freescale P2020RDB (2.6.32 kernel) SELinux (2.6.32 kernel) Wind River Linux 3 Yellow Dog™ Linux 4.0	Table 4.3 on page 16
LynxOS®[a]	LynxOS 4.0, 4.2, 5.0	Table 5.1 on page 24

Table 1.1 **Platforms Available with Release 5.0**

Operating System		Reference
Mac OS®	Mac OS X 10.6	Table 6.1 on page 28
QNX®	QNX Neutrino® 6.4.1, 6.5	Table 7.1 on page 32
Solaris™	Solaris 2.9, 2.10	Table 8.1 on page 37
VxWorks®	VxWorks 5.5, 6.3 - 6.9 VxWorks 653 2.3 VxWorks MILS 2.1.1	Table 9.1 on page 43
Windows®	Windows 7 (32-bit and 64-bit Editions) Windows 2003 (32-bit and 64-bit Editions) Windows Server 2008 R2 (64-bit Edition) Windows Vista® (32-bit and 64-bit Editions) Windows XP Professional (32-bit and 64-bit Editions) with Service Pack 2	Table 10.1 on page 57

[a] The Java API is not supported on LynxOS platforms in 5.0.0. If your application requires support for Java on LynxOS, please contact your RTI account manager.

For each platform, this document provides information on:

- □ Supported operating systems and compilers
- □ Required *Connext* and system libraries
- □ Required compiler and linker flags
- □ Required environment variables for running the application (if any)
- □ Details on how the *Connext* libraries were built
- □ Multicast support
- □ Supported transports
- □ Monotonic clock support
- □ CPU core affinity control support

2 AIX Platforms

Table 2.1 on page 4 lists the architectures supported on the IBM® AIX operating system.

Table 2.2 on page 5 lists the compiler flags and the libraries you will need to link into your application.

Table 2.3 on page 6 provides details on the environment variables required to be set at run time for an AIX architecture.

Table 2.4 on page 6 provides details on how the libraries were built. This table is provided strictly for informational purposes; you do not need to use these parameters to compile your application. You may find this information useful if you are involved in any in-depth debugging.

2.1 Changing Thread Priority

Due to the AIX threading-model implementation, there are situations that require you to run your *Connext* application with root privileges:

❑ **For all APIs:** Your application must have *root* privileges to use the thread option, DDS_THREAD_SETTINGS_REALTIME_PRIORITY, for the event and receiver pool thread QoS (**DDS_DomainParticipantQos.event.thread, DDS_DomainParticipantQos.receiver_pool.thread**).

❑ **For the Java API only:** Your application must have *root* privileges to change the event and receiver pool thread priorities **(DDS_DomainParticipantQos.event.thread, DDS_DomainParticipantQos.receiver_pool.thread**).

2.2 Multicast Support

Multicast is supported on all AIX platforms and is configured out of the box. That is, the default value for the initial peers list (NDDS_DISCOVERY_PEERS) includes a multicast address. See the online documentation for more information.

2.3 Supported Transports

Shared memory: Supported and enabled by default.

UDPv4: Supported and enabled by default.

UDPv6: Not supported.

TCP/IPv4: Not supported.

2.3.1 Notes for Using Shared Memory

By default, the maximum number of shared memory segments you can use with AIX is quite small and limits the capability of *Connext* applications to work properly over shared memory. To increase the maximum number of shared memory segments an application can use, set the following environment variable before invoking your *Connext* application:

```
EXTSHM=ON
```

This environment variable is not required if your application does not use the shared memory transport.

To see a list of shared memory resources in use, please use the '**ipcs**' command. To clean up shared memory and shared semaphore resources, please use the '**ipcrm**' command.

The shared memory keys used by *Connext* are in the range of 0x400000. For example:

```
ipcs -m | grep 0x004
```

The shared semaphore keys used by *Connext* are in the range of 0x800000; the shared mutex keys are in the range of 0xb00000. For example:

```
ipcs -s | grep 0x008
ipcs -s | grep 0x00b
```

Please refer to the shared-memory transport online documentation for details on the shared memory and semaphore keys used by *Connext*.

2.4 Monotonic Clock Support

The monotonic clock (described in Section 8.6 in the *RTI Core Libraries and Utilities User's Manual*) is not supported on AIX platforms.

2.5 Support for Controlling CPU Core Affinity for RTI Threads

Support for controlling CPU core affinity (described in Section 19.5 in the *RTI Core Libraries and Utilities User's Manual*) is not available for AIX platforms.

Table 2.1 **Supported AIX Target Platforms**

Operating System	CPU	Compiler	RTI Architecture Abbreviation
AIX 5.3	POWER5 (32-bit mode)	IBM XLC for AIX v9.0	p5AIX5.3xlc9.0
		IBM Java 1.6	p5AIX5.3xlc9.0jdk
	POWER5 (64-bit mode)	IBM XLC for AIX v9.0	64p5AIX5.3xlc9.0
		IBM Java 1.6	64p5AIX5.3xlc9.0jdk

Table 2.2 **Building Instructions for AIX Architectures**

API	Library Format	Required RTI Libraries[a]	Required System Libraries[b]	Required Compiler Flags
C++	Static Release	libnddscppz.a libnddscz.a libnddscorez.a For *Connext Messaging*, also include: librticonnextmsgcppz.a	-ldl -lnsl -lm -pthread	-DRTI_AIX -DRTI_UNIX -q[32 | 64] [c] -qlongdouble
	Static Debug	libnddscppzd.a libnddsczd.a libnddscorezd.a For *Connext Messaging*, also include: librticonnextmsgcppzd.a		
	Dynamic Release	libnddscpp.so libnddsc.so libnddscore.so For *Connext Messaging*, also include: librticonnextmsgcpp.so	-ldl -lnsl -lm -pthread -brtl	
	Dynamic Debug	libnddscppd.so libnddscd.so libnddscored.so For *Connext Messaging*, also include: librticonnextmsgcppd.so		
C	Static Release	libnddscz.a libnddscorez.a For *Connext Messaging*, also include: librticonnextmsgcz.a	-ldl -lnsl -lm -pthread	-DRTI_AIX -DRTI_UNIX -q[32 | 64] [c] -qlongdouble -qthreaded [d]
	Static Debug	libnddsczd.a libnddscorezd.a For *Connext Messaging*, also include: librticonnextmsgczd.a		
	Dynamic Release	libnddsc.so libnddscore.so For *Connext Messaging*, also include: librticonnextmsgc.so	-ldl -lnsl -lm -pthread -brtl	
	Dynamic Debug	libnddscd.so libnddscored.so For *Connext Messaging*, also include: librticonnextmsgcd.so		
Java	Release	nddsjava.jar For *Connext Messaging*, also include: rticonnextmsg.jar	N/A	N/A
	Debug	nddsjavad.jar For *Connext Messaging*, also include: rticonnextmsgd.jar		

[a] The *Connext* C/C++ libraries are located in $(NDDSHOME)/lib/<*architecture*>/.
(where $(NDDSHOME) is where *Connext* is installed, such as /local/rti/ndds.5.0.x)

[b] Transports (other than the default IP transport) such as StarFabric may require linking in additional libraries. For further details, see the online documentation or contact support@rti.com.

[c] Use '-q32' if you build 32-bit code or '-q64' for 64-bit code.

d. The '-qthreaded' option is automatically set if you use one of the compilers that ends with '_r', such as cc_r, xlc_r, xlC_r. See the IBM XLC reference manual for more information.

Table 2.3 **Running Instructions for AIX Architectures**

RTI Architecture	Library Format *(Release & Debug)*	Required Environment Variables

p5AIX5.3xlc9.0jdk	N/A	LD_LIBRARY_PATH=$(NDDSHOME)/lib/<*arch*>: $(LD_LIBRARY_PATH)[a]
		EXTSHM=ON[b]

64p5AIX5.3xlc9.0jdk		
All other supported architectures	Static	EXTSHM=ON[b]

	Dynamic	LD_LIBRARY_PATH=$(NDDSHOME)/lib/<*arch*>: $(LD_LIBRARY_PATH)
		EXTSHM=ON[b]

a. ${NDDSHOME} represents the root directory of your *Connext* installation. ${LD_LIBRARY_PATH} represents the value of the LD_LIBRARY_PATH variable prior to changing it to support *Connext*. When using nddsjava.jar, the Java virtual machine (JVM) will attempt to load release versions of the native libraries (nddsjava.so, nddscore.so, nddsc.so). When using nddsjavad.jar, the JVM will attempt to load debug versions of the native libraries (nddsjava.so, nddscore.so, nddsc.so).

b. See Notes for Using Shared Memory (Section 2.3.1).

Table 2.4 **Library-Creation Details for AIX Architectures**

RTI Architecture	Library Format *(Static & Dynamic)*	Compiler Flags Used by RTI [a]
p5AIX5.3xlc9.0	Release	-q32 -qlongdouble -qalias=noansi -qpic=large -qthreaded -D_POSIX_C_SOURCE=199506L -D__EXTENSIONS__ -O -qflag=i:i -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=Power5+ -DNDEBUG
	Debug	-q32 -qlongdouble -qalias=noansi -qpic=large -qthreaded -D_POSIX_C_SOURCE=199506L -D__EXTENSIONS__ -O -qflag=i:i -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=Power5+ -g
64p5AIX5.3xlc9.0	Release	-q64 -qwarn64 -qlongdouble -qalias=noansi -qpic=large -qthreaded -D_POSIX_C_SOURCE=199506L -D__EXTENSIONS__ -O -qflag=i:i -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=Power5+ -DNDEBUG
	Debug	-q64 -qwarn64 -qlongdouble -qalias=noansi -qpic=large -qthreaded -D_POSIX_C_SOURCE=199506L -D__EXTENSIONS__ -O -qflag=i:i -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=Power5+ -g
All supported AIX architectures for Java	Release	-target 1.4 -source 1.4
	Debug	-target 1.4 -source 1.4 -g

a. *Connext* was built using the 'xlC_r' compiler. See IBM's XLC reference manual for a description of the different compilers. For a list of the additional settings (defined by default) for the 'xlC_r' compiler, see the file, **/etc/vac.cfg.53**.

3 INTEGRITY Platforms

Table 3.1 lists the architectures supported on the INTEGRITY® operating system.

Table 3.1 **Supported INTEGRITY Target Platforms**[a]

Operating System	CPU	Compiler	IP Stack	RTI Architecture Abbreviation
INTEGRITY 5.0.11	PPC 85XX	multi 4.2.4	GHnet2 IP stack[b]	ppc85xxInty5.0.11.xes-p2020
INTEGRITY 10.0.2 [c]	x86	multi 5.0.6	CHNet IPv4 stack	pentiumInty10.0.2.pcx86

[a] For use with Windows and Solaris hosts, as supported by Green Hills Software.

[b] Kernel must be built using -lip4 or -lip46.

[c] Requires patch_6901.iff from Green Hills Software.

Table 3.2 lists the compiler flags and the libraries you will need to link into your application.

Table 3.2 **Building Instructions for INTEGRITY Architectures**

API	Library Format	Required RTI Libraries[a]	Required System Libraries[b]	Required Compiler Flags
C++	Static Release	libnddscppz.a libnddscz.a libnddscorez.a For *Connext Messaging*, also include: librticonnextmsgcppz.a	libsocket.a libnet.a libposix.a	RTI_INTY --exceptions
	Static Debug	libnddscppzd.a libnddsczd.a libnddscorezd.a (libnddscppzd.dba)[c] (libnddsczd.dba)[c] (libnddscorezd.dba)[c] For *Connext Messaging*, also include: librticonnextmsgcppzd.a		
C	Static Release	libnddscz.a libnddscorez.a For *Connext Messaging*, also include: librticonnextmsgcz.a		
	Static Debug	libnddsczd.a libnddscorezd.a (libnddsczd.dba)[c] (libnddscorezd.dba)[c] For *Connext Messaging*, also include: librticonnextmsgczd.a		

[a] The *Connext* C/C++ libraries are located in $(NDDSHOME)/lib/<*architecture*>/.
(where $(NDDSHOME) is where *Connext* is installed, such as /local/rti/ndds.5.0.x)

[b] Transports (other than the default IP transport) such as StarFabric may require linking in additional libraries. For further details, see the online documentation or contact support@rti.com.

[c] The *.dba files contain the debugging information. You can link without these, as long as they are located in the same directory as the matching *d.a file (so that the MULTI® IDE can find the debug information).

Table 3.3 provides details on the environment variables required to be set at run time for an INTEGRITY architecture.

Table 3.3 **Running Instructions for INTEGRITY Architectures**

RTI Architecture	Required Environment Variables
All INTEGRITY architectures	None

Table 3.4 provides details on how the libraries were built. This table is provided strictly for informational purposes; you do not need to use these parameters to compile your application. You may find this information useful if you are involved in any in-depth debugging.

Table 3.4 **Library-Creation Details for INTEGRITY Architectures**

RTI Architecture	Library Format	Compiler Flags Used by RTI
pentiumInty10.0.2.pcx86	Static Release	-bspname=pcx86 -prefixed_msgs --unknown_pragma_silent -G -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU= -DTARGET=\"pentiumInty10.0.2.pcx86\" -DNDEBUG -c
	Static Debug	-bspname=pcx86 -prefixed_msgs --unknown_pragma_silent -G -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU= -DTARGET=\"pentiumInty10.0.2.pcx86\" -c
ppc85xxInty5.0.11.xes-p2020	Static Release	-bspname=xes-p2020 -prefixed_msgs --unknown_pragma_silent -G -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU= -DTARGET=\"ppc85xxInty5.0.11.xes-p2020\" -DNDEBUG -c
	Static Debug	-bspname=xes-p2020 -prefixed_msgs --unknown_pragma_silent -G -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU= -DTARGET=\"ppc85xxInty5.0.11.xes-p2020\" -c

3.1 Patch Required for INTEGRITY 10.02 Platform

To run a *Connext* application on an INTEGRITY 10.0.2 system, you must install patch_6901.iff from Green Hills Software. For more information, please contact your Green Hills Software representative.

3.2 Support for Request-Reply Communication Pattern

RTI Connext Messaging includes support for the Request-Reply Communication Pattern for the platforms described in Table 3.1 and all programming languages, except as noted below.

When using C++, the following platform does not support the Request-Reply Communication Pattern:

❏ ppc85xxInty5.0.11.xes-p2020

3.3 Diagnostics on INTEGRITY Systems

Connext libraries for the INTEGRITY platforms use **consolestring()**, which prints debugging information to the serial console when available. Using the serial console as opposed to the target I/O window (host I/O) is generally recommended. Host I/O will affect the real-time performance of the target. For more information on **consolestring()**, please refer to the *INTEGRITY Development Guide.*

3.4 Socket-Enabled and POSIX-Enabled Threads are Required

Connext on INTEGRITY platforms internally relies on the POSIX API for many of its system calls. As a result, any thread calling *Connext* must be POSIX-enabled. By default, the 'Initial' thread of an address space is POSIX-enabled, provided the address space has been linked with **libposix.a**. Additional user threads that call *Connext* must be spawned from the Initial thread using **pthread_create**. Only then is the created thread also POSIX-enabled. Note that tasks created at build time using the Integrate utility are *not* POSIX-enabled.

Furthermore, threads calling *Connext* must be socket-enabled. This can be achieved by calling **InitLibSocket()** before making any *Connext* calls and calling **ShutdownLibSocket** before the thread terminates. Note that an Initial thread is, by default, socket-enabled when the address space is linked with **libsocket.a**. Please refer to the *INTEGRITY Development Guide* for more information.

3.5 Running over IP Backplane on a Dy4 Champ-AVII Board

Connext can run on all four CPUs, provided the following hold true:

❏ *Connext* applications on CPUs B, C and D only exchange data with applications on a different CPU or off-board.

❏ The IP backplane and associated routing has been properly configured. *Connext* has been tested with the following libraries built into the INTEGRITY kernel: **debug**, **res**, **load**, **socket**, **itcpip**, **lbp**, **queue**, **ifbp**, **idb**, **bsl**.

3.6 Multi-NIC Support on INTEGRITY 5.0

Due to limitations with the API of the InterPeak stack for INTEGRITY 5.0, *Connext* only supports a single NIC when the InterPeak stack is used. This NIC must be called **"eth0"**. By default on an INTEGRITY system, this will correspond to the first network card, which can be changed by reconfiguring the kernel. This limitation does not affect the InterNiche stack.

3.7 Multicast Support

Multicast is supported on all INTEGRITY 5.0 and 10.0 platforms.

3.8 Supported Transports

Shared memory: Supported, enabled by default. To clean up shared memory resources, reboot the kernel.

UDPv4: Supported, enabled by default.

UDPv6: Not supported.

TCP/IPv4: Not supported.

3.8.1 Smaller Shared-Memory Receive-Resource Queue Size

INTEGRITY's shared-memory pluggable transport uses the shared-memory POSIX API. This API is part of the standard INTEGRITY distribution and is shipped as a library. The current version (5.0.4) of this library uses a hard-coded value for the total amount of memory that can be shared with an address space. This limits the overall buffer space that can be used by the *DomainParticipants* within the same address space to communicate over shared memory with other *DomainParticipants*.

To allow more *DomainParticipants* to run within the same address space, we reduced the default size of the queue for each receive resource of the shared memory transport. The queue size is reduced to 8 messages (the default for other platforms is 32). This change only applies to INTEGRITY architectures and this default value can be overwritten through the shared memory transport QoS.

3.8.2 Using Shared Memory on INTEGRITY Systems

Connext uses the single address-space POSIX library to implement the shared-memory transport on INTEGRITY 10.0 operating systems.

To use shared-memory, you must configure your system to include the POSIX shared-memory library. The **posix_shm_manager** must be running in an "AddressSpace" solely dedicated to it. After building any *Connext* application that uses shared memory, you must use the **intex** utility (provided with the INTEGRITY development environment) to pack the application with multiple address-spaces: one (or more) to contain the *Connext* application(s), and another one to contain the **posix_shm_manager**.

Connext will run on a target without the **posix_shm_manager**, but the POSIX functions will fail and return **ENOSYS**, and the participants will fail to communicate through shared memory.

To include the POSIX Shared-Memory Manager in its own Address Space:

The project files generated by *rtiddsgen* for MULTI will create the shared-memory manager for you. Please follow these steps:

1. Specify the path to your INTEGRITY distribution in the **_default.gpj** top-level project file by adding the following line (modify it according to the path to your INTEGRITY distribution):

   ```
   -os_dir=/local/applications/integrity/integrity-10.0.2
   ```

2. Build the project.
3. Before running your *Connext* application on a target, download the **posix_shm_manager** file (generated by the build) onto the target.

The POSIX Shared Memory Manager will start automatically after the download and your applications will be able to use shared memory.

Notes:

- ❏ Only *one* **posix_shm_manager** is needed on a particular target. INTEGRITY offers the option of building this **posix_shm_manager** *inside* the kernel. Please refer to the INTEGRITY documentation.
- ❏ If you are already using shared memory through the POSIX library, there may be a possible conflict.
- ❏ INTEGRITY 5 has two different types of POSIX library: a single-address space one (or 'light') and another one (complete POSIX implementation). *Connext* uses the first one, but will work if you are using the complete POSIX implementation.

3.8.3 Shared Memory Limitations on INTEGRITY Systems

If several applications are running on the same INTEGRITY node and are using shared memory, once an application is stopped, it cannot be restarted. When the application is stopped (gracefully or ungracefully), any new application on the same domain index within the same domain will fail to start until the shared memory manager is also restarted.

Additionally, if the application is stopped ungracefully, the remaining applications will print several error messages such as the following until *Connext* purges the stopped application from its database:

```
Resource Manager send error = 0x9
```

This error message is logged from INTEGRITY's POSIX shared memory manager, *not* from *Connext.* The error message is benign and will not prevent the remaining applications from communicating with each other or with application on other nodes.

The workaround is to either restart the stopped application with a different participant index or shut down all the other applications and the shared memory manager, then restart everything.

3.9 Using rtiddsping and rtiddsspy on PowerPC INTEGRITY Systems

While the RTI libraries for INTEGRITY can be used with any BSP, providing the PowerPC processor falls under the same category (for example, the ppc7400... RTI libraries can be used on any target with a PPC74xx processor), *rtiddsping* and *rtiddsspy* are provided as executables, and therefore are BSP-dependent. You will not be able to run them successfully on your target if it is not compatible with the BSP listed in the architecture name (such as mvme5100-7400). Please refer to your hardware documentation for peripheral compatibility across BSPs.

3.10 Monotonic Clock Support

The monotonic clock (described in Section 8.6 in the *RTI Core Libraries and Utilities User's Manual*) is not supported.

3.11 Support for Controlling CPU Core Affinity for RTI Threads

Support for controlling CPU core affinity (described in Section 19.5 in the *RTI Core Libraries and Utilities User's Manual*) is not available for INTEGRITY platforms.

3.12 Issues with INTEGRITY Systems

3.12.1 Delay When Writing to Unreachable Peers

On INTEGRITY systems, if a publishing application's initial peers list includes a nonexistent (or simply unreachable) host, calls to **write()** may block for approximately 1 second.

This long block is caused by the stack trying to resolve the invalid/unreachable host. Most IP stacks do not block the sending thread because of this reason, and you may include invalid/ unreachable hosts in your initial-peers list. If you find that your stack does block the sending thread, please consult your IP stack vendor on how to change its behavior. [RTI Issue ID CORE-1637, Bug # 10768]

3.12.2 Linking with 'libivfs.a' without a File System

If you link your application with **libivfs.a** and are using a system that does not have a file system, you may notice the application blocks for 2 seconds at start-up.

3.12.3 Compiler Warnings Regarding Unrecognized #pragma Directives

Building *Connext* projects for INTEGRITY causes the compiler to produce several warnings about #pragma directives not recognized in some *Connext* header files. For example:

```
Building default.bld
"C:/ndds/ndds.4.4x/include/ndds/dds_c/dds_c_infrastructure.h", line 926:
warning: unrecognized #pragma
  #pragma warning(push)
          ^
"C:/ndds/ndds.4.4x/include/ndds/dds_c/dds_c_infrastructure.h", line 927:
warning: unrecognized #pragma
  #pragma warning(disable:4190)
          ^
"C:/ndds/ndds.4.4x/include/ndds/dds_c/dds_c_infrastructure.h", line 945:
warning: unrecognized #pragma
  #pragma warning(pop)
          ^
```

These warnings do not compromise the final application produced and can be safely ignored.

3.12.4 Warning when Loading Connext Applications on INTEGRITY Systems

When a *Connext* application compiled with the *rtiddsgen*-generated project files is loaded on an INTEGRITY 5.0.x target, the following warning appears:

```
"Warning: Program is linked with libc.so POSIX signals and cancellation
will not work."
```

The *Connext* libraries do not use the additional features provided by the full POSIX implementation, therefore the warning can safely be ignored. This warning is due to the fact that the *rtiddsgen*-generated project files use the Single AddressSpace POSIX library by default, not the full POSIX implementation on INTEGRITY (POSIX System). The *Connext* libraries only require Single AddressSpace POSIX to function correctly, but will still work if you are using the POSIX System. The message indicates that items such as inter-process signaling or process-shared semaphores will not be available (more information can be found in the *INTEGRITY Libraries and Utilities User's Guide*, chapter "Introduction to POSIX on INTEGRITY").

4 Linux and Fedora Platforms

First, see the basic instructions for compiling on Linux platforms provided in Section 9.3 in the *RTI Core Libraries and Utilities User's Manual*. The following tables provide supplemental information.

Table 4.1 on page 15 through Table 4.3 on page 16 list the supported Linux and Fedora architectures.

Table 4.4 on page 17 lists the compiler flags and libraries you will need to link into your application. See also: Monotonic Clock Support (Section 4.4).

Table 4.5 on page 18 provides details on the environment variables required to be set at run time for a Linux architecture. When running on 64-bit Java architectures (x64Linux2.6...jdk), use the **-d64** flag in the command-line.

Table 4.6 on page 18 provides details on how the Linux libraries were built. This table is provided strictly for informational purposes; you do not need to use these parameters to compile your application. You may find this information useful if you are involved in any in-depth debugging.

Table 4.7 on page 21 and Table 4.8 on page 22 list additional libraries required when using the optional *RTI Secure WAN Transport* and *RTI TCP Transport*, respectively.

4.1 Native POSIX Thread Library (NPTL) Requirements

This section applies only to these architectures:

❑ **Yellow Dog Linux 4.0:** ppc7400Linux2.6gcc3.3.3

To use the above architectures, you must have the development version of Native POSIX Thread Library (NPTL) installed on your host system, and the NPTL libraries on your target system.

❑ When you *build* the application, you must have the development NPTL library installed in **/usr/lib/nptl**. This library is not installed by default.

❑ To see if your system has NPTL installed, look for this directory: **/usr/lib/nptl**. It should contain these files: **libpthread.so** and **libpthread.a**.

If NPTL is not installed, you will need to install a package that includes it, such as **nptl-devel**. This package is not typically part of a default installation. You can find it either in your original Linux installation media (CD/DVD) or, if you have upgraded your system, through the distribution's update site.

❑ When you *run* the application, it will automatically use the default NPTL library in **/lib/nptl**. You do not need the development library installed on the target system.

Note: Make sure the environment variable LD_ASSUME_KERNEL is either not defined at all, or is set to 2.4.20 or higher. The middleware will not run if it is set to less than 2.4.20.[1]

[1] The dynamic loader (ld), is configured by default to load the NPTL library, as long as LD_ASSUME_KERNEL is NOT defined.

4.2 Multicast Support

Multicast is supported on all Linux and Fedora platforms and is configured out of the box. That is, the default value for the initial peers list (**NDDS_DISCOVERY_PEERS**) includes a multicast address. See the online documentation for more information.

4.3 Supported Transports

Shared memory: Supported and enabled by default. To clean up shared memory resources, reboot the kernel.

UDPv4: Supported and enabled by default.

UDPv6: Supported for all platforms listed in Table 4.1 on page 15 through Table 4.3 on page 16 *except* SELinux 2.6.32 kernel (ppc4xxFPLinux2.6gcc4.5.1).

The UDPv6 transport is not enabled by default, and the peers list must be modified to support IPv6.

Note: Traffic Class support is only provided on architectures with gcc 4.1.0 or later that support the UDPv6 transport.

TCP/IPv4: Supported on CentOS 5.4 and 5.5, Red Hat Enterprise Linux 4.0 and higher (except Red Hat Enterprise Linux 5.2 with Real-Time Extensions), and Ubuntu Server 10.04. (This is *not* a built-in transport.)

4.3.1 Shared Memory Support

To see a list of shared memory resources in use, please use the '**ipcs**' command. To clean up shared memory and shared semaphore resources, please use the '**ipcrm**' command.

The shared memory keys used by *Connext* are in the range of 0x400000. For example:

```
ipcs -m | grep 0x004
```

The shared semaphore keys used by *Connext* are in the range of 0x800000; the shared mutex keys are in the range of 0xb00000. For example:

```
ipcs -s | grep 0x008
ipcs -s | grep 0x00b
```

Please refer to the shared-memory transport online documentation for details on the shared memory and semaphore keys used by *Connext*.

4.4 Monotonic Clock Support

The monotonic clock (described in Section 8.6 in the *RTI Core Libraries and Utilities User's Manual*) is supported on all Linux, SUSE, and Fedora 2.6 kernel platforms.

4.5 Support for Controlling CPU Core Affinity for RTI Threads

Support for controlling CPU core affinity (described in Section 19.5 in the *RTI Core Libraries and Utilities User's Manual*) is available on all supported Linux, SUSE, and Fedora platforms.

Note: The API for controlling CPU core affinity may change in future releases.

4.6 Libraries Required for Using RTI Secure WAN Transport APIs

This section is only relevant if you have installed *RTI Secure WAN Transport.* This feature is not part of the *RTI Core Libraries and Utilities*. If you choose to use it, it must be downloaded and installed separately. It is only available on specific architectures. See the *RTI Secure WAN Transport Release Notes* and *RTI Secure WAN Transport Release Notes Installation Guide* for details.

To use the WAN or Secure Transport APIs, link against the additional libraries from Table 4.7 on page 21. (Select the files appropriate for your chosen library format.)

4.7 Libraries Required for Using RTI TCP Transport APIs

To use the TCP Transport APIs, link against the additional libraries from Table 4.8 on page 22. If you are using *RTI TLS Support*, see Table 4.9 on page 22. (Select the files appropriate for your chosen library format.)

Table 4.1 **Linux Platforms on Cell BE CPUs**

Operating System	CPU	Compiler	RTI Architecture Abbreviation
Fedora 12 (2.6.32 kernel)	Cell BE	gcc 4.5.1[a], glib 2.9	cell64Linux2.6gcc4.5.1

[a] Requires a custom version of gcc 4.5.1.

Table 4.2 **Linux Platforms on Intel CPUs**

Operating System	CPU	Compiler	RTI Architecture Abbreviation
CentOS 5.4, 5.5 (2.6 kernel)	x86	gcc 4.1.2	i86Linux2.6gcc4.1.2
		Sun Java Platform Standard Edition JDK 1.6	i86Linux2.6gcc4.1.2jdk
	x64	gcc 4.1.2	x64Linux2.6gcc4.1.2
		Sun Java Platform Standard Edition JDK 1.6	x64Linux2.6gcc4.1.2jdk
CentOS 6.0	x86	gcc 4.4.5	i86Linux2.6gcc4.4.5
		Sun Java Platform Standard Edition JDK 1.6	i86Linux2.6gcc4.4.5jdk
	x64	gcc 4.4.5	x64Linux2.6gcc4.4.5
		Sun Java Platform Standard Edition JDK 1.6	x64Linux2.6gcc4.4.5jdk
Fedora 12 (2.6.32 kernel)	x64	gcc 4.4.4	x64Linux2.6gcc4.4.4
Fedora 12 (2.6.32 kernel) with gcc 4.5.1	x64	gcc 4.5.1[a]	x64Linux2.6gcc4.5.1
Red Hat Enterprise Linux 5.0 (2.6 kernel)	x86	gcc 4.1.1	i86Linux2.6gcc4.1.1
		Sun Java Platform Standard Edition JDK 1.6	i86Linux2.6gcc4.1.1jdk
	x64	gcc 4.1.1	x64Linux2.6gcc4.1.1
		Sun Java Platform Standard Edition JDK 1.6	x64Linux2.6gcc4.1.1jdk
Red Hat Enterprise Linux 5.1, 5.2, 5.4, 5.5 (2.6 kernel)	x86	gcc 4.1.2	i86Linux2.6gcc4.1.2
		Sun Java Platform Standard Edition JDK 1.6	i86Linux2.6gcc4.1.2jdk
	x64	gcc 4.1.2	x64Linux2.6gcc4.1.2
		Sun Java Platform Standard Edition JDK 1.6	x64Linux2.6gcc4.1.2jdk

Table 4.2 **Linux Platforms on Intel CPUs**

Operating System	CPU	Compiler	RTI Architecture Abbreviation
Red Hat Enterprise Linux 5.2 with Real-Time Extensions (2.6 kernel)	x86	gcc 4.1.2	i86Linux2.6gcc4.1.2
		Sun Java Platform Standard Edition JDK 1.6	i86Linux2.6gcc4.1.2jdk
Red Hat Enterprise Linux 6.0, 6.1 (2.6 kernel)	x86	gcc 4.4.5	i86Linux2.6gcc4.4.5
		Sun Java Platform Standard Edition JDK 1.6	i86Linux2.6gcc4.4.5jdk
	x64	gcc 4.4.5	x64Linux2.6gcc4.4.5
		Sun Java Platform Standard Edition JDK 1.6	x64Linux2.6gcc4.4.5jdk
SUSE Linux Enterprise Server 10.1 (2.6 kernel)	x86	gcc 4.1.0	i86Suse10.1gcc4.1.0
		Sun Java Platform Standard Edition JDK 1.5 and 1.6	i86Suse10.1gcc4.1.0jdk
	x64	gcc 4.1.0	x64Suse10.1gcc4.1.0
		Sun Java Platform Standard Edition JDK 1.5 and 1.6	x64Suse10.1gcc4.1.0jdk
Ubuntu Server 10.04 (LTS)	x86	gcc 4.4.3	i86Linux2.6gcc4.4.3
		Sun Java Platform Standard Edition JDK 1.6	i86Linux2.6gcc4.4.3jdk
	x64	gcc 4.4.3	x64Linux2.6gcc4.4.3
		Sun Java Platform Standard Edition JDK 1.6	x64Linux2.6gcc4.4.3jdk
Wind River Linux 4 (2.6 kernel)	x64	gcc 4.4.1	x64WRLinux2.6gcc4.4.1

a. Requires a custom version of gcc 4.5.1.

Table 4.3 **Linux Platforms on PowerPC CPUs**

Operating System	CPU	Compiler	RTI Architecture Abbreviation
Freescale P2020RDB (2.6.32 kernel)	PPC 85xx	Freescale gcc.4.3.74 based on gcc.4.3.2	ppc85xxLinux2.6gcc4.3.2
SELinux (2.6.32 kernel)	PPC 4xxFP	gcc 4.5.1[a], glibc 2.9	ppc4xxFPLinux2.6gcc4.5.1
Wind River Linux 3	PPC 85xx	gcc 4.3.2	ppc85xxWRLinux2.6gcc4.3.2
Yellow Dog® Linux 4.0 (2.6 kernel)	PPC 74xx (such as 7410)	gcc 3.3.3	ppc7400Linux2.6gcc3.3.3

a. Requires a custom version of gcc 4.5.1.

Table 4.4 **Building Instructions for Linux and Fedora Architectures**

API	Library Format	Required RTI Libraries or Jar Files[a]	Required System Libraries	Required Compiler Flags
C++	Static Release	libnddscppz.a libnddscz.a libnddscorez.a For *Connext Messaging*, also include: librticonnextmsgcz.a	For all *Linux2.6gcc3* architectures: -ldl -lnsl -lm -L/usr/lib/nptl -lpthread -lrt All other Linux architectures: -ldl -lnsl -lm -lpthread -lrt	64-bit architectures: -DRTI_UNIX -m64 32-bit architectures: -DRTI_UNIX -m32
	Static Debug	libnddscppzd.a libnddsczd.a libnddscorezd.a For *Connext Messaging*, also include: librticonnextmsgcppzd.a		
	Dynamic Release	libnddscpp.so libnddsc.so libnddscore.so For *Connext Messaging*, also include: librticonnextmsgcpp.so		
	Dynamic Debug	libnddscppd.so libnddscd.so libnddscored.so For *Connext Messaging*, also include: librticonnextmsgcppd.so		
C	Static Release	libnddscz.a libnddscorez.a For *Connext Messaging*, also include: librticonnextmsgcz.a	For all *Linux2.6gcc3* architectures: -ldl -lnsl -lm -L/usr/lib/nptl -lpthread -lrt All other Linux architectures: -ldl -lnsl -lm -lpthread -lrt	64-bit architectures: -DRTI_UNIX -m64 32-bit architectures: -DRTI_UNIX -m32
	Static Debug	libnddsczd.a libnddscorezd.a For *Connext Messaging*, also include: librticonnextmsgczd.a		
	Dynamic Release	libnddsc.so libnddscore.so For *Connext Messaging*, also include: librticonnextmsgc.so		
	Dynamic Debug	libnddscd.so libnddscored.so For *Connext Messaging*, also include: librticonnextmsgcd.so		
Java	Release	nddsjava.jar For *Connext Messaging*, also include: rticonnextmsg.jar	N/A	None required
	Debug	nddsjavad.jar For *Connext Messaging*, also include: rticonnextmsgd.jar		

[a] RTI C/C++ libraries are in $(NDDSHOME)/lib/<*architecture*>/. RTI Java files are in $(NDDSHOME)/class/ (where $(NDDSHOME) is where *Connext* is installed, such as /local/rti/ndds.5.0.x).

Table 4.5 **Running Instructions for Linux and Fedora Architectures**

RTI Architecture	Library Format	Environment Variables
All supported Linux/ SUSE/Fedora architectures for Java	N/A	LD_LIBRARY_PATH= ${NDDSHOME}/lib/<*architecture*>: ${LD_LIBRARY_PATH}[a] **Note:** For all 64-bit Java architectures (...64Linux...jdk), use **-d64** in the command line.
All other supported Linux/SUSE/Fedora architectures	Static (Release & Debug)	None required
	Dynamic (Release & Debug)	LD_LIBRARY_PATH= ${NDDSHOME}/lib/<*architecture*>: ${LD_LIBRARY_PATH}[a]

[a] ${NDDSHOME} represents the root directory of your *Connext* installation. ${LD_LIBRARY_PATH} represents the value of the LD_LIBRARY_PATH variable prior to changing it to support *Connext*. When using nddsjava.jar, the Java virtual machine (JVM) will attempt to load release versions of the native libraries. When using nddsjavad.jar, the JVM will attempt to load debug versions of the native libraries.

Table 4.6 **Library-Creation Details for Linux and Fedora Architectures**

RTI Architecture	Library Format *(Static & Dynamic)*	Compiler Flags Used by RTI
cell64Linux2.6gcc4.5.1	Release	-O3 -fPIC -mminimal-toc -mcpu=cell -mtune=cell -DLINUX -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=cell -DTARGET=\"cell64Linux2.6gcc4.5.1\" -DNDEBUG -c -Wp,-MD
	Debug	-O3 -fPIC -mminimal-toc -mcpu=cell -mtune=cell -DLINUX -g -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=cell -DTARGET=\"cell64Linux2.6gcc4.5.1\" -c -Wp,-MD
i86Linux2.6gcc4.1.1	Release	-fPIC -DLINUX -O -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=I80586 -DTARGET=\"i86Linux2.6gcc4.1.1\" -fmessage-length=0 -DNDEBUG -c -Wp,-MD
	Debug	-fPIC -DLINUX -g -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=I80586 -DTARGET=\"i86Linux2.6gcc4.1.1\" -fmessage-length=0 -c -Wp,-MD
i86Linux2.6gcc4.1.2	Release	-fPIC -DLINUX -O -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=I80586 -DTARGET=\"i86Linux2.6gcc4.1.2\" -fmessage-length=0 -DNDEBUG -c -Wp,-MD
	Debug	-fPIC -DLINUX -g -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=I80586 -DTARGET=\"i86Linux2.6gcc4.1.2\" -fmessage-length=0 -c -Wp,-MD
i86Linux2.6gcc4.4.3	Release	-fPIC -DLINUX -O -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=I80586 -DTARGET=\"i86Linux2.6gcc4.4.3\" -fmessage-length=0 -DNDEBUG -c -Wp,-MD
	Debug	-fPIC -DLINUX -g -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=I80586 -DTARGET=\"i86Linux2.6gcc4.4.3\" -fmessage-length=0 -c -Wp,-MD

Table 4.6 **Library-Creation Details for Linux and Fedora Architectures**

RTI Architecture	Library Format *(Static & Dynamic)*	Compiler Flags Used by RTI
i86Linux2.6gcc4.4.5	Release	gcc -m32 -fPIC -DLINUX -O -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=I80586 -DTARGET=\\"i86Linux2.6gcc4.4.5\\" -DNDEBUG -Wp,-MD
	Debug	gcc -m32 -fPIC -DLINUX -g -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=I80586 -DTARGET=\\"i86Linux2.6gcc4.4.5\\" -Wp,-MD
i86Suse10.1gcc4.1.0	Release	-fPIC -DLINUX -O -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=I80586 -DTARGET=\"i86Suse10.1gcc4.1.0\" -fmessage-length=0 -DNDEBUG -c -Wp,-MD
	Debug	-fPIC -DLINUX -g -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=I80586 -DTARGET=\"i86Suse10.1gcc4.1.0\" -fmessage-length=0 -c -Wp,-MD
ppc7400Linux2.6gcc3.3.3 [a]	Release	-fPIC -DLINUX -O -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=PPC7400 -DTARGET=\"ppc7400Linux2.6gcc3.3.3\" -DNDEBUG -c -Wp,-MD
	Debug	-fPIC -DLINUX -g -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=PPC7400 -DTARGET=\"ppc7400Linux2.6gcc3.3.3\" -c -Wp,-MD
ppc4xxFPLinux2.6gcc4.3.3	Release	-fPIC -DLINUX -O -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=4xxFP -DTARGET=\"ppc4xxFPLinux2.6gcc4.3.3\" -DNDEBUG -c -Wp,-MD
	Debug	-fPIC -DLINUX -g -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=4xxFP -DTARGET=\"ppc4xxFPLinux2.6gcc4.3.3\" -DNDEBUG -c -Wp,-MD
ppc4xxFPLinux2.6gcc4.5.1	Release	-fPIC -DLINUX -O -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=4xxFP -DTARGET=\"ppc4xxFPLinux2.6gcc4.5.1\" -DNDEBUG -c -Wp,-MD
	Debug	-fPIC -DLINUX -g -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=4xxFP -DTARGET=\"ppc4xxFPLinux2.6gcc4.5.1\" -DNDEBUG -c -Wp,-MD
ppc85xxLinux2.6gcc4.3.2	Release	-fPIC -DLINUX -g -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntyType=long -DCREAL_IS_FLOAT -DCPU=e500 -DTARGET=\"ppc85xxLinux2.6gcc4.3.2\" -DNDEBUG -c -Wp, -MD
	Debug	-fPIC -DLINUX -g -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntyType=long -DCREAL_IS_FLOAT -DCPU=e500 -DTARGET=\"ppc85xxLinux2.6gcc4.3.2\" -c -Wp, -MD
ppc85xxWRLinux2.6gcc4.3.2	Release	-mcpu=powerpc -msoft-float -fPIC -DLINUX -O -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=PPC32 -DTARGET=\"ppc85xxWRLinux2.6gcc4.3.2\" -DNDEBUG -Wp,-MD
	Debug	powerpc-wrs-linux-gnu-gcc -mcpu=powerpc -msoft-float -fPIC -DLINUX -g -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=PPC32 -DTARGET=\"ppc85xxWRLinux2.6gcc4.3.2\" -Wp,-MD

Table 4.6 **Library-Creation Details for Linux and Fedora Architectures**

RTI Architecture	Library Format *(Static & Dynamic)*	Compiler Flags Used by RTI
x64Linux2.6gcc4.1.1[a]	Release	-m64 -fPIC -DLINUX -O -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=AMD64 -DTARGET=\"x64Linux2.6gcc4.1.1\" -DNDEBUG -c -Wp,-MD
	Debug	-m64 -fPIC -DLINUX -g -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=AMD64 -DTARGET=\"x64Linux2.6gcc4.1.1\" -fmessage-length=0 -c -Wp,-MD
x64Linux2.6gcc4.1.2 [a]	Release	-m64 -fPIC -DLINUX -O -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=AMD64 -DTARGET=\"x64Linux2.6gcc4.1.2\" -DNDEBUG -c -Wp,-MD
	Debug	-m64 -fPIC -DLINUX -g -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=AMD64 -DTARGET=\"x64Linux2.6gcc4.1.2\" -fmessage-length=0 -c -Wp,-MD
x64Linux2.6gcc4.3.4	Release	-m64 -fPIC -DLINUX -O -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=AMD64 -DTARGET=\"x64Linux2.6gcc\" -c -Wp,-MD
	Debug	-m64 -fPIC -DLINUX -g -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=AMD64 -DTARGET=\"x64Linux2.6gcc4.3.4\" -c -Wp,-MD
x64Linux2.6gcc4.4.4 [a]	Release	-m64 -fPIC -DLINUX -O -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=AMD64 -DTARGET=\"x64Linux2.6gcc4.4.4\" -DNDEBUG -c -Wp,-MD
	Debug	-m64 -fPIC -DLINUX -g -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=AMD64 -DTARGET=\"x64Linux2.6gcc4.4.4\" -c -Wp,-MD
x64Linux2.6gcc4.4.3	Release	-m64 -fPIC -DLINUX -O -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=AMD64 -DTARGET=\"x64Linux2.6gcc4.4.3\" -DNDEBUG -c -Wp,-MD
	Debug	-m64 -fPIC -DLINUX -g -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=AMD64 -DTARGET=\"x64Linux2.6gcc4.4.3\" -fmessage-length=0 -c -Wp,-MD
x64Linux2.6gcc4.4.5	Release	gcc -m64 -fPIC -DLINUX -O -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=AMD64 -DTARGET=\\\"x64Linux2.6gcc4.4.5\\\" -DNDEBUG -Wp,-MD
	Debug	gcc -m64 -fPIC -DLINUX -g -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=AMD64 -DTARGET=\\\"x64Linux2.6gcc4.4.5\\\" -Wp,-MD

Table 4.6 **Library-Creation Details for Linux and Fedora Architectures**

RTI Architecture	Library Format *(Static & Dynamic)*	Compiler Flags Used by RTI
x64Linux2.6gcc4.5.1	Release	-fPIC -DLINUX -O -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=AMD64 -DTARGET=\"x64Linux2.6gcc4.5.1\" -DNDEBUG -c -Wp,-MD
	Debug	-fPIC -DLINUX -g -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=AMD64 -DTARGET=\"x64Linux2.6gcc4.5.1\" -fmessage-length=0 -c -Wp,-MD
x64Suse10.1gcc4.1.0	Release	-m64 -fPIC -DLINUX -O -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=AMD64 -DTARGET=\"x64Suse10.1gcc4.1.0\" -DNDEBUG -c -Wp,-MD
	Debug	-m64 -fPIC -DLINUX -g -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=AMD64 -DTARGET=\"x64Suse10.1gcc4.1.0\" -fmessage-length=0 -c -Wp,-MD
x64WRLinux2.6gcc4.4.1	Release	-m64 -march=x86-64 -mtune=generic -fPIC -DLINUX -O -Wall -Wno-unknown-pragmas -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=AMD64 -DNDEBUG
	Debug	-m64 -march=x86-64 -mtune=generic -fPIC -DLINUX -O -Wall -Wno-unknown-pragmas -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=AMD64 -DNDEBUG
All supported Linux and Fedora architectures for Java (*jdk)	Dynamic Release	-target 1.4 -source 1.4
	Dynamic Debug	-target 1.4 -source 1.4 -g

a. The C++ libnddscpp dynamic libraries were linked using g++; the C dynamic libraries, i.e., libnddscore and libnddsc, were linked using gcc.

Table 4.7 **Additional Libraries for using RTI Secure WAN Transport APIs on UNIX-based Systems**

Library Format	RTI Secure WAN Transport Libraries[a]	OpenSSL Libraries[b]
Dynamic Release	libnddstransportwan.so libnddstransporttls.so	libssl.so libcrypto.so
Dynamic Debug	libnddstransportwand.so libnddstransporttlsd.so	
Static Release	libnddstransporttlsz.a libnddstransporttlszd.a	
Static Debug	libnddstransportwanz.a libnddstransportwanzd.a	

a. The libraries are located in <*wan install dir*>/lib/<*architecture*>/., where <*wan install dir*> is where you installed *RTI Secure WAN Transport*, such as /local/rti/ndds.5.0.x.

b. These libraries are located <*openssl install dir*>/<*architecture*/lib, where <*openssl install dir*> is where you installed OpenSSL, such as /local/rti/openssl-0.9.8x.

Table 4.8 **Additional Libraries for using RTI TCP Transport APIs on UNIX-based Systems**

Library Format	RTI TCP Transport Libraries[a]
Dynamic Release	libnddstransporttcp.so
Dynamic Debug	libnddstransporttcpd.so
Static Release	libnddstransporttcpz.a
Static Debug	libnddstransporttcpzd.a

a. The libraries are located in <*Connext install dir*>/lib/<*architecture*>/., where <*Connext install dir*> is where you installed *Connext*, such as /local/rti/ndds.5.0.x.

Table 4.9 **Additional Libraries for using RTI TCP Transport APIs on UNIX-based Systems with TLS Enabled**

Library Format	RTI TLS Libraries[a]
Dynamic Release	libnddstls.so
Dynamic Debug	libnddstlsd.so
Static Release	libnddstlsz.a
Static Debug	libnddstlszd.a
OpenSSL Libraries	libssl.so libcrypto.so

a. The libraries are located in <*TLS install dir*>/lib/<*architecture*>/., where <*TLS install dir*> is where you installed *RTI TLS Support*, such as /local/rti/ndds.5.0.x.

5 LynxOS Platforms

Table 5.1 on page 24 lists the architectures supported on LynxOS® operating systems.

Table 5.2 on page 24 and Table 5.3 on page 25 list the compiler flags and libraries you will need to link into your application.

Table 5.4 on page 25 provides details on the environment variables required to be set at run time for a LynxOS architecture.

Table 5.5 on page 25 provides details on how the libraries were built by RTI. This table is provided strictly for informational purposes; you do not need to use these parameters to compile your application. You may find this information useful if you are involved in any in-depth debugging.

Note: The Java API is not supported on LynxOS platforms in *Connext* 5.0.0. If you would like Java to be supported on LynxOS, please contact your RTI account manager.

5.1 Multicast Support

Multicast is supported on all LynxOS platforms, but it is not configured out of the box. That is, the default value for the initial peers list (NDDS_DISCOVERY_PEERS) does not include a multicast address.

To configure a LynxOS target to use multicast, you need to add routes so multicast packets will be sent via the proper network interfaces. To add routes, use the "route add" command. The specific parameters depend on how the target is configured, the name of the interface (such as **elxl0** in the example below), etc. Please refer to your LynxOS documentation for details on the "route add" command.

For example:

```
route add -net 224.0.0.0 -netmask 240.0.0.0 -interface elxl0
```

Note—Group Address Ignored for Multicast Reception on Loopback: On LynxOS architectures, the multicast-loopback implementation ignores the group address when receiving messages. This causes *Connext* to receive all outgoing multicast traffic originating from the host for that port. Thus, if you have two participants on the same host and in the same domain, both listening for discovery traffic over multicast, they will discover each other, regardless of the multicast address to which they are listening. (The correct behavior would be to receive messages only for the addresses to which the current process (not the host) is subscribed.)

5.2 Supported Transports

Shared memory: Supported and enabled by default.

UDPv4: Supported and enabled by default.

UDPv6: Not supported.

TCP/IPv4: Not supported.

5.2.1 Shared Memory Support

To see a list of shared memory resources in use, use the '**ipcs**' command. To clean up shared memory and shared semaphore resources, use the '**ipcrm**' command.

The shared memory keys used by *Connext* are in the range of 0x400000. For example:

```
ipcs -m | grep 0x004
```

The shared semaphore keys used by *Connext* are in the range of 0x800000; the shared mutex keys are in the range of 0xb00000. For example:

```
ipcs -s | grep 0x008
ipcs -s | grep 0x00b
```

Please refer to the shared-memory transport online documentation for details on the shared memory and semaphore keys used by *Connext*.

5.3 Monotonic Clock Support

The monotonic clock (described in Section 8.6 in the *RTI Core Libraries and Utilities User's Manual*) is not supported on LynxOS platforms.

5.4 Support for Controlling CPU Core Affinity for RTI Threads

Support for controlling CPU core affinity (described in Section 19.5 in the *RTI Core Libraries and Utilities User's Manual*) is not available for LynxOS platforms.

Table 5.1 **Supported LynxOS Platforms**

Operating System	CPU	Compiler	RTI Architecture
LynxOS 4.0	x86	gcc 3.2.2	i86Lynx4.0.0gcc3.2.2
	PPC 74xx (such as 7410)	gcc 3.2.2	ppc7400Lynx4.0.0gcc3.2.2
	PPC 604, PPC 7XX (such as 750)	gcc 3.2.2	ppc750Lynx4.0.0gcc3.2.2
LynxOS 4.2	PPC 74xx (such as 7410)	gcc 3.2.2	ppc7400Lynx4.2.0gcc3.2.2
LynxOS 5.0	PPC 74xx (such as 7410)	gcc 3.4.3	ppc7400Lynx5.0.0gcc3.4.3

Table 5.2 **Building Instructions for LynxOS Architectures (Connext Libraries)**

API	Library Format[a]	Required RTI Libraries [b]
C++	Static Release	libnddscppz.a libnddscz.a libnddscorez.a For *Connext Messaging*, also include: librticonnextmsgcppz.a
	Static Debug	libnddscppzd.a libnddsczd.a libnddscorezd.a For *Connext Messaging*, also include: librticonnextmsgcppzd.a
	Dynamic Release	libnddscpp.so libnddsc.so libnddscore.so For *Connext Messaging*, also include: librticonnextmsgcpp.so
	Dynamic Debug	libnddscppd.so libnddscd.so libnddscored.so For *Connext Messaging*, also include: librticonnextmsgcppd.so

Table 5.2 **Building Instructions for LynxOS Architectures (Connext Libraries)**

API	Library Format[a]	Required RTI Libraries [b]
C	Static Release	libnddscz.a libnddscorez.a For *Connext Messaging*, also include: librticonnextmsgcz.a
	Static Debug	libnddsczd.a libnddscorezd.a For *Connext Messaging*, also include: librticonnextmsgczd.a
	Dynamic Release	libnddsc.so libnddscore.so For *Connext Messaging*, also include: librticonnextmsgc.so
	Dynamic Debug	libnddscd.so libnddscored.so For *Connext Messaging*, also include: librticonnextmsgcd.so

[a] Dynamic libraries are not supported under LynxOS-178.

[b] The *Connext* C/C++ libraries are located in $(NDDSHOME)\lib\<*architecture*>\. (where $(NDDSHOME) is where *Connext* is installed, such as c:\rti\ndds.5.0.x)

Table 5.3 **Building Instructions for LynxOS Architectures (System Libraries and Compiler Flags)**

API	RTI Architecture	Required System Libraries	Required Compiler Flags
C and C++	i86Lynx4.0.0gcc3.2.2	-ldb -lm -lrpc -lc -llynx	-DRTI_LYNX -mthreads -mshared
	ppc7400Lynx4.0.0gcc3.2.2		
	ppc7400Lynx4.2.0gcc3.2.2		
	ppc7400Lynx5.0.0gcc3.4.3		
	ppc750Lynx4.0.0gcc3.2.2		

Table 5.4 **Running Instructions for LynxOS Architectures**

RTI Architecture	Library Format *(Release & Debug)*	Required Environment Variables
All supported LynxOS architectures	Static	None required
	Dynamic	LD_LIBRARY_PATH= ${NDDSHOME}/lib/<*architecture*>: ${LD_LIBRARY_PATH}

Table 5.5 **Library-Creation Details for LynxOS Architectures**

RTI Architecture	Library Format *(Static & Dynamic)*	Compiler Flags Used by RTI
i86Lynx4.0.0gcc3.2.2	Release	-mthreads -mshared -fPIC -D_POSIX_THREADS_CALLS -D__NO_INCLUDE_WARN__ -O -Wall -Wno-unknown-pragmas -DPtrIntType=long -DCPU=I80586 -DTARGET=\"i86Lynx4.0.0gcc3.2.2\" -DNDEBUG -c -Wp,-MD
	Debug	-mthreads -mshared -fPIC -D_POSIX_THREADS_CALLS -D__NO_INCLUDE_WARN__ -g -O -Wall -Wno-unknown-pragmas -DPtrIntType=long -DCPU=I80586 -DTARGET=\"i86Lynx4.0.0gcc3.2.2\" -c -Wp,-MD

Table 5.5 **Library-Creation Details for LynxOS Architectures**

RTI Architecture	Library Format *(Static & Dynamic)*	Compiler Flags Used by RTI
ppc7400Lynx4.0.0gcc3.2.2	Release	-mcpu=7400 -maltivec -mabi=altivec -fno-exceptions -mthreads -mshared -fPIC -D_POSIX_THREADS_CALLS -D__NO_INCLUDE_WARN__ -O -Wall -Wno-unknown-pragmas -DPtrIntType=long -DCPU=PPC7400 -DTARGET=\"ppc7400Lynx4.0.0gcc3.2.2\" -DNDEBUG -c -Wp,-MD
	Debug	-mcpu=7400 -maltivec -mabi=altivec -fno-exceptions -mthreads -mshared -fPIC -D_POSIX_THREADS_CALLS -D__NO_INCLUDE_WARN__ -g -O -Wall -Wno-unknown-pragmas -DPtrIntType=long -DCPU=PPC7400 -DTARGET=\"ppc7400Lynx4.0.0gcc3.2.2\" -c -Wp,-MD
ppc7400Lynx4.2.0gcc3.2.2	Release	-mcpu=7400 -maltivec -mabi=altivec -fno-exceptions -mthreads -mshared -fPIC -D_POSIX_THREADS_CALLS -D__NO_INCLUDE_WARN__ -O -Wall -Wno-unknown-pragmas -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=PPC7400 -DTARGET=\"ppc7400Lynx4.2.0gcc3.2.2\" -DNDEBUG -c -Wp,-MD
	Debug	-mcpu=7400 -maltivec -mabi=altivec -fno-exceptions -mthreads -mshared -fPIC -D_POSIX_THREADS_CALLS -D__NO_INCLUDE_WARN__ -O -Wall -Wno-unknown-pragmas -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=PPC7400 -DTARGET=\"ppc7400Lynx4.2.0gcc3.2.2\" -c -Wp,-MD
ppc7400Lynx5.0.0gcc3.4.3	Release	-mcpu=7400 -maltivec -mabi=altivec -fno-exceptions -mthreads -mshared -fPIC -D_POSIX_THREADS_CALLS -D__NO_INCLUDE_WARN__ -O -Wall -Wno-unknown-pragmas -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=PPC7400 -DTARGET=\"ppc7400Lynx5.0.0gcc3.4.3\" -DNDEBUG -c -Wp,-MD
	Debug	-mcpu=7400 -maltivec -mabi=altivec -fno-exceptions -mthreads -mshared -fPIC -D_POSIX_THREADS_CALLS -D__NO_INCLUDE_WARN__ -O -Wall -Wno-unknown-pragmas -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=PPC7400 -DTARGET=\"ppc7400Lynx5.0.0gcc3.4.3\" -c -Wp,-MD
ppc750Lynx4.0.0gcc3.2.2	Release	-mcpu=750 -fno-exceptions -mthreads -mshared -fPIC -D_POSIX_THREADS_CALLS -D__NO_INCLUDE_WARN__ -O -Wall -Wno-unknown-pragmas -DPtrIntType=long -DCPU=PPC750 -DTARGET=\"ppc750Lynx4.0.0gcc3.2.2\" -DNDEBUG -c -Wp,-MD
	Debug	-mcpu=750 -fno-exceptions -mthreads -mshared -fPIC -D_POSIX_THREADS_CALLS -D__NO_INCLUDE_WARN__ -g -O -Wall -Wno-unknown-pragmas -DPtrIntType=long -DCPU=PPC750 -DTARGET=\"ppc750Lynx4.0.0gcc3.2.2\" -c -Wp,-MD

6 Mac OS Platforms

Table 6.1 on page 28 lists the architectures supported on Mac OS operating systems.

Table 6.2 on page 28 lists the compiler flags and libraries you will need to link into your application.

Table 6.3 on page 29 provides details on the environment variables required to be set at run time for a Mac OS architecture.

Table 6.4 on page 30 provides details on how the libraries were built by RTI. This table is provided strictly for informational purposes; you do not need to use these parameters to compile your application. You may find this information useful if you are involved in any in-depth debugging.

To use *rtiddsgen* to create files for a Java target:

By default, *rtiddsgen* uses JRE 1.4, which is distributed with the *RTI Core Libraries and Utilities* in **$NDDSHOME/jre**. To use *rtiddsgen* with Java 1.6, you need Xalan-Java 2.7.1 or higher.

1. Download the latest Xalan-Java version from http://xml.apache.org/xalan-j/.
2. Unpack the Xalan distribution file in a location of your choice.
3. Set the XALANHOME environment variable to the Xalan installation directory.
4. Set the NDDSJREHOME environment variable to the JRE 1.6 installation directory.
5. Run *rtiddsgen* using the script located in **$NDDSHOME/scripts**.

6.1 Multicast Support

Multicast is supported on Mac OS platforms and is configured out of the box. That is, the default value for the initial peers list (**NDDS_DISCOVERY_PEERS**) includes a multicast address. See the online documentation for more information.

6.2 Supported Transports

Shared memory: Supported and enabled by default.

UDPv4: Supported and enabled by default.

UDPv6: Not supported.

TCP/IPv4: Not supported.

6.3 Monotonic Clock Support

The monotonic clock (described in Section 8.6 in the *RTI Core Libraries and Utilities User's Manual*) is not supported on Mac OS platforms.

6.4 Support for Controlling CPU Core Affinity for RTI Threads

Support for controlling CPU core affinity (described in Section 19.5 in the *RTI Core Libraries and Utilities User's Manual*) is not available for Mac OS platforms.

Table 6.1 **Mac OS Platforms**

Operating System	CPU	Compiler	RTI Architecture Abbreviation
Mac OS X 10.6	x64	gcc 4.2.1	x64Darwin10gcc4.2.1
		Java SE 1.6 for Mac OS	x64Darwin10gcc4.2.1jdk

Table 6.2 **Building Instructions for Mac OS Architectures**

API	Library Format	Required RTI Libraries[a]	Required System Libraries	Required Compiler Flags
C++	Static Release	libnddscppz.a libnddscz.a libnddscorez.a For *Connext Messaging*, also include: librticonnextmsgcppz.a	-ldl -lm -lpthread	-dynamic -lpthread -lc -single_module
	Static Debug	libnddscppzd.a libnddsczd.a libnddscorezd.a For *Connext Messaging*, also include: librticonnextmsgcppzd.a		
	Dynamic Release	libnddscpp.dylib libnddsc.dylib libnddscore.dylib For *Connext Messaging*, also include: librticonnextmsgcpp.dylib		
	Dynamic Debug	libnddscppd.dylib libnddscd.dylib libnddscored.dylib For *Connext Messaging*, also include: librticonnextmsgcppd.dylib		

Table 6.2 **Building Instructions for Mac OS Architectures**

API	Library Format	Required RTI Libraries[a]	Required System Libraries	Required Compiler Flags
C	Static Release	libnddscz.a libnddscorez.a For *Connext Messaging*, also include: librticonnextmsgcz.a	-ldl -lm -lpthread	-dynamic -lpthread -lc -single_module
	Static Debug	libnddsczd.a libnddscorezd.a For *Connext Messaging*, also include: librticonnextmsgczd.a		
	Dynamic Release	libnddsc.dylib libnddscore.dylib For *Connext Messaging*, also include: librticonnextmsgc.dylib		
	Dynamic Debug	libnddscd.dylib libnddscored.dylib For *Connext Messaging*, also include: librticonnextmsgcd.dylib		
Java	Release	nddsjava.jar For *Connext Messaging*, also include: rticonnextmsg.jar	N/A	None required
	Debug	nddsjavad.jar For *Connext Messaging*, also include: rticonnextmsgd.jar		

[a] The Connext C/C++ libraries are located in $(NDDSHOME)/lib/<architecture>/. (where $(NDDSHOME) is where *Connext* is installed, such as /local/rti/ndds.5.0.x)

Table 6.3 **Running Instructions for Mac OS Architectures**

RTI Architecture	Library Format *(Release & Debug)*	Required Environment Variables
x64Darwin10gcc4.2.1	Static	None required
	Dynamic	DYLD_LIBRARY_PATH=${NDDSHOME}/lib/ x64Darwin10gcc4.2.1:${DYLD_LIBRARY_PATH}[a]
x64Darwin10gcc4.2.1jdk	N/A	DYLD_LIBRARY_PATH=${NDDSHOME}/lib/ x64Darwin10gcc4.2.1jdk:${DYLD_LIBRARY_PATH}[a]

[a] ${NDDSHOME} represents the root directory of your *Connext* installation. ${LD_LIBRARY_PATH} represents the value of the LD_LIBRARY_PATH variable prior to changing it to support *Connext*. When using nddsjava.jar, the Java virtual machine (JVM) will attempt to load release versions of the native libraries (nddsjava.dylib, nddscore.dylib, nddsc.dylib). When using nddsjavad.jar, the JVM will attempt to load debug versions of the native libraries (nddsjava.dylib, nddscore.dylib, nddsc.dylib).

Table 6.4 **Library-Creation Details for Mac OS Architectures**

RTI Architecture	Library Format *(Static & Dynamic)*	Compiler Flags Used by RTI
x64Darwin10gcc4.2.1	Release	-O -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=AMD64 -DTARGET=\"x64Darwin10gcc4.2.1\" -c -Wp,-MD
	Debug	-g -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=AMD64 -DTARGET=\" x64Darwin10gcc4.2.1\" -c -Wp,-MD
x64Darwin10gcc4.2.1jdk	Release	-target 1.4 -source 1.4
	Debug	-target 1.4 -source 1.4 -g

7 QNX Platforms

Table 7.1 on page 32 lists the architectures supported on QNX operating systems.

Table 7.2 on page 33 lists the libraries you will need to link into your application.

Table 7.3 on page 34 provides details on the environment variables required to be set at run time for a QNX architecture.

Table 7.4 on page 34 provides details on how the QNX libraries were built.

7.1 Required Change for Building with C++ Libraries for QNX Platforms—New in 5.0.0

For QNX architectures, starting in release 5.0.0: The C++ libraries are now built *without* the **-fno-rtti** flag and *with* the **-fexceptions** flag. To build QNX architectures starting with release 5.0.0, you must build your C++ applications *without* **-fno-exceptions** in order to link with the RTI libraries. In summary:

- ❑ Do *not* use **-fno-exceptions** when building a C++ application or the build will fail. It is not necessary to use -fexceptions, but doing so will not cause a problem.
- ❑ It is no longer necessary to use **-fno-rtti**, but doing so will not cause a problem.

7.2 Multicast Support

Multicast is supported on QNX platforms and is configured out of the box. That is, the default value for the initial peers list (NDDS_DISCOVERY_PEERS) includes a multicast address. See the online documentation for more information.

7.3 Supported Transports

Shared Memory: Supported and enabled by default.

To see a list of the shared memory resources, enter:

```
'ls /dev/shmem/RTIOsapiSharedMemorySegment-*'
```

To clean up the shared memory resources, remove the files listed in **dev/shmem/**. The shared resource names used by *Connext* begin with **'RTIOsapiSharedMemorySem-'**. To see a list of shared semaphores, enter:

```
'ls /dev/sem/RTIOsapiSharedMemorySemMutex*'
```

To clean up the shared semaphore resources, remove the files listed in **/dev/sem/**'.

The permissions for the semaphores created by *Connext* are modified by the process' **umask** value. If you want to have shared memory support between different users, run the command "**umask 000**" to change the default **umask** value to 0 before running your *Connext* application.

UDPv4: Supported and enabled by default.

UDPv6: Supported. The transport is not enabled by default; the peers list must be modified to support IPv6. No Traffic Class support.

To use the UDPv6 transport, the network stack must provide IPv6 capability. Enabling UDPv6 may involve switching the network stack server and setting up IPv6 route entries.

TCP/IPv4: Supported on QNX 6.5 platforms.

TLS: Supported on QNX 6.5 platforms.

7.4 Monotonic Clock Support

The monotonic clock (described in Section 8.6 in the *RTI Core Libraries and Utilities User's Manual*) is supported on QNX platforms.

7.5 Support for Controlling CPU Core Affinity for RTI Threads

Support for controlling CPU core affinity (described in Section 19.5 in the *RTI Core Libraries and Utilities User's Manual*) is not available for QNX platforms.

7.6 Restarting Applications on QNX Systems

Due to a limitation in the POSIX API, if a process is unexpectedly interrupted in the middle of a critical section of code that is protected by a shared mutex semaphore, the OS is not able to automatically release the semaphore, making it impossible to reuse it by another application.

The *Connext* shared-memory transport uses a shared mutex to protect access to the shared memory area across multiple processes.

It is possible under some extreme circumstances that if one application crashes or terminates ungracefully while executing code inside a critical section, the other applications sharing the same resource will not be able to continue their execution. If this situation occurs, you must manually delete the shared-memory mutex before re-launching any application in the same domain.

Table 7.1 **Supported QNX Platforms**[a]

Operating System	CPU	Compiler	RTI Architecture
QNX Neutrino 6.4.1	Pentium class	qcc 4.3.3 with GNU C++ libraries	i86QNX6.4.1qcc_gpp
QNX Neutrino 6.5	Pentium class	qcc 4.4.2 with GNU C++ libraries	i86QNX6.5qcc_gpp4.4.2

[a] For use with Windows, Linux or Solaris Host as supported by QNX & RTI

Table 7.2 **Building Instructions for QNX Architectures**

API	Library Format	RTI Libraries[a]	Required System Libraries	Required Compiler Flags
C++	Static Release	libnddscppz.a libnddscz.a libnddscorez.a For *Connext Messaging*, also include: librticonnextmsgcppz.a	-lm -lsocket	-DRTI_QNX
	Static Debug	libnddscppzd.a libnddsczd.a libnddscorezd.a For *Connext Messaging*, also include: librticonnextmsgcppzd.a		
	Dynamic Release	libnddscpp.so libnddsc.so libnddscore.so For *Connext Messaging*, also include: librticonnextmsgcpp.so		
	Dynamic Debug	libnddscppd.so libnddscd.so libnddscored.so For *Connext Messaging*, also include: librticonnextmsgcppd.so		
C	Static Release	libnddscz.a libnddscorez.a For *Connext Messaging*, also include: librticonnextmsgcz.a	-lm -lsocket	-DRTI_QNX
	Static Debug	libnddsczd.a libnddscorezd.a For *Connext Messaging*, also include: librticonnextmsgczd.a		
	Dynamic Release	libnddsc.so libnddscore.so For *Connext Messaging*, also include: librticonnextmsgc.so		
	Dynamic Debug	libnddscd.so libnddscored.so For *Connext Messaging*, also include: librticonnextmsgcd.so		

[a] The NDDS C/C++ libraries are located in $(NDDSHOME)\lib\<architecture>\. (where $(NDDSHOME) is where *Connext* is installed, such as c:\rti\ndds.5.0.x)

Table 7.3 **Running Instructions for QNX Architectures**

RTI Architecture	Library Format *(Release & Debug)*	Environment Variables
i86QNX6.4.1qcc_gpp, i86QNX6.5qcc_gpp4.4.2	Static	None required
	Dynamic	LD_LIBRARY_PATH= ${NDDSHOME}/lib/<*architecture*>: ${LD_LIBRARY_PATH}[a]

[a] ${NDDSHOME} represents the root directory of your *Connext* installation. ${LD_LIBRARY_PATH} represents the value of the LD_LIBRARY_PATH variable prior to changing it to support *Connext*. When using nddsjava.jar, the Java virtual machine (JVM) will attempt to load release versions of the native libraries. When using nddsjavad.jar, the JVM will attempt to load debug versions of the native libraries.

Table 7.4 **Library-Creation Details for QNX Architectures**

RTI Architecture	Library Format *(Static & Dynamic)*	Compiler Flags Used by RTI
i86QNX6.4.1qcc_gpp	Release	qcc -Vgcc/4.3.3,gcc_ntox86 -Y_gpp -lang-c -fPIC -fexceptions -O -Wall -Wno-unknown-pragmas -DNDEBUG
	Debug	qcc -Vgcc/4.3.3,gcc_ntox86 -Y_gpp -lang-c -fPIC -fexceptions -g -Wall -Wno-unknown-pragmas
i86QNX6.5qcc_gpp4.4.2	Release	qcc -Vgcc/4.4.2,gcc_ntox86 -Y_gpp -m32 -march=i386 -mtune=generic -fPIC -fexceptions -DFD_SETSIZE=512 -O -Wall -Wno-unknown-pragmas -DRTS_QNX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=I80586 -DTARGET=\"i86QNX6.5qcc_gpp4.4.2\" -DNDEBUG
	Debug	qcc -Vgcc/4.4.2,gcc_ntox86 -Y_gpp -m32 -march=i386 -mtune=generic -fPIC -fexceptions -DFD_SETSIZE=512 -g -Wall -Wno-unknown-pragmas -DRTS_QNX-DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=I80586 -DTARGET=\"i86QNX6.5qcc_gpp4.4.2\"

8 Solaris Platforms

Table 8.1 on page 37 lists the architectures supported on Solaris operating systems.

Table 8.2 on page 38 lists the compiler flags and the libraries you will need to link into your application. (See also: Libraries Required for using RTI Secure WAN Transport APIs (Section 8.6).)

Table 8.3 on page 39 provides details on the environment variables required to be set at run time for a Solaris architecture.

When running on a Java 64-bit architecture, use the **-d64** flag in the command-line.

Table 8.4 on page 39 provides details on how the libraries were built by RTI. This table is provided strictly for informational purposes; you do not need to use these parameters to compile your application. You may find this information useful if you are involved in any in-depth debugging.

8.1 Support for Request-Reply Communication Pattern

RTI Connext Messaging includes support for the Request-Reply Communication Pattern, for all the platforms described in Table 8.1 on page 37 and all programming languages, except as noted below.

When using C++, the following platforms do not support the Request-Reply Communication Pattern:

❏ sparcSol2.9cc5.4

8.2 Multicast Support

Multicast is supported on Solaris platforms and is configured out of the box. That is, the default value for the initial peers list (**NDDS_DISCOVERY_PEERS**) includes a multicast address. See the online documentation for more information.

8.3 Supported Transports

Shared memory: Supported and enabled by default.

UDPv4: Supported and enabled by default.

UDPv6: Supported for all Solaris 2.9 and 2.10 platforms. The transport is not enabled by default, and the peers list must be modified to support IPv6. Traffic Class support is only provided for Solaris 2.10 platforms.

TCP/IPv4: Not supported.

8.3.1 Shared Memory Support

To see a list of shared memory resources in use, use the '**ipcs**' command. To clean up shared memory and shared semaphore resources, use the '**ipcrm**' command.

The shared memory keys used by *Connext* are in the range of 0x400000. For example:

```
ipcs -m | grep 0x4
```

The shared semaphore keys used by *Connext* are in the range of 0x800000; the shared mutex keys are in the range of 0xb00000. For example:

```
ipcs -s | grep 0x8
ipcs -s | grep 0xb
```

Please refer to the shared-memory transport online documentation for details on the shared memory and semaphore keys used by *Connext*.

8.3.2 Increasing Available Shared Resources

Connext uses System V semaphores to manage shared memory communication. If you plan to run multiple *Connext* applications on the same node, at the same time, you may need to increase the number of available semaphores.

Each *Connext* application that has shared memory enabled allocates 4 individual semaphores. The Solaris system defaults allow only 10 per host, which may not be enough (one is often used by the system, so you'll run out at the 3rd application).

To increase the number of semaphores available to *Connext*, change the values of the following two parameters in **/etc/system**. (Starting in Solaris 10, there is an alternate mechanism to control these values, but changing **/etc/system** will also work.) The following values are just an example:

```
set semsys:seminfo_semmni = 100
set semsys:seminfo_semmns = 100
```

If these parameters already exist in **/etc/system**, change their values; otherwise, add the above lines to your **/etc/system** file.

❑ **WARNING:** Changing **/etc/system** should be done VERY carefully—incorrect editing of the file can render your system unbootable!

"System V" semaphores are allocated by creating groups of individual semaphores. The first parameter above controls the maximum number of semaphore groups and the second controls the maximum total number of semaphores (within any and all groups). Each *Connext* application that has shared memory enabled allocates 4 groups of 1 semaphore each (per domain). So setting the two values to be the same number will work fine as far as *Connext* is concerned. However, if other applications in the system want to allocate bigger groups, you could set "semsys:seminfo_semmns" larger than "semsys:seminfo_semmni." (Setting semmni bigger than semmns does not make any sense, since groups can't have less than 1 semaphore.)

In the absence of other applications using them, having 100 System V semaphores will allow you to use 25 domain ID/participant index combinations for *Connext* applications. You probably will not need to increase the shared memory parameters, since the default allows 100 shared memory areas, enough for 50 applications.

8.4 Monotonic Clock Support

The monotonic clock (described in Section 8.6 in the *RTI Core Libraries and Utilities User's Manual*) is supported on all Solaris architectures.

8.5 Support for Controlling CPU Core Affinity for RTI Threads

Support for controlling CPU core affinity (described in Section 19.5 in the *RTI Core Libraries and Utilities User's Manual*) is not available for Solaris platforms.

8.6 Libraries Required for using RTI Secure WAN Transport APIs

This section is only relevant if you have installed *RTI Secure WAN Transport.* This feature is not part of the standard *Connext* package. If you choose to use it, it must be downloaded and installed separately. It is only available on specific architectures. See the *RTI Secure WAN Transport Release Notes* and *RTI Secure WAN Transport Release Notes Installation Guide* for details. To use the WAN or Secure Transport APIs, link against the additional libraries from Table 8.7 on page 40. (Select the files appropriate for your chosen library format.)

Table 8.1 **Supported Solaris Platforms**

Operating System	CPU	Compiler or Software Development Kit	RTI Architecture
Solaris 2.9	x86	gcc 3.3.2	i86Sol2.9gcc3.3.2
		Sun Java Platform Standard Edition JDK 1.6	i86Sol2.9jdk
	UltraSPARC	CC 5.4 (Forte Dev 7, Sun One Studio 7)	sparcSol2.9cc5.4
		Sun Java Platform Standard Edition JDK 1.6	sparcSol2.9jdk
Solaris 10	UltraSPARC	gcc3.4.2	sparcSol2.10gcc3.4.2
		Sun Java Platform Standard Edition JDK 1.5 or 1.6	sparcSol2.10jdk
	UltraSPARC (with native 64-bit support)	gcc3.4.2	sparc64Sol2.10gcc3.4.2
		Sun Java Platform Standard Edition JDK 1.5 or 1.6	sparc64Sol2.10jdk

Table 8.2 **Building Instructions for Solaris Architectures**

API	Library Format	RTI Libraries or Jar Files[a]	Required System Libraries	Required Compiler Flags
C	Static Release	libnddscz.a libnddscorez.a For *Connext Messaging*, also include: librticonnextmsgcz.a	sparc64Sol2.10gcc3.4.2: -ldl -lnsl -lsocket -lgen -lposix4 -lpthread -lm -lc All other architectures: -ldl -lnsl -lgenIO -lsocket -lgen -lposix4 -lpthread -lm -lc	sparc64Sol2.10gcc3.4.2: -DRTI_UNIX -m64 All other architectures: -DRTI_UNIX -m32
	Static Debug	libnddsczd.a libnddscorezd.a For *Connext Messaging*, also include: librticonnextmsgczd.a		
	Dynamic Release	libnddsc.so libnddscore.so For *Connext Messaging*, also include: librticonnextmsgc.so		
	Dynamic Debug	libnddscd.so libnddscored.so For *Connext Messaging*, also include: librticonnextmsgcd.so		
C++	Static Release	libnddscppz.a libnddscz.a libnddscorez.a For *Connext Messaging*, also include: librticonnextmsgcppz.a	sparc64Sol2.10gcc3.4.2: -ldl -lnsl -lsocket -lgen -lposix4 -lpthread -lm -lc All other architectures: -ldl -lnsl -lgenIO -lsocket -lgen -lposix4 -lpthread -lm -lc	sparc64Sol2.10gcc3.4.2: -DRTI_UNIX -m64 All other architectures: -DRTI_UNIX -m32
	Static Debug	libnddscppzd.a libnddsczd.a libnddscorezd.a For *Connext Messaging*, also include: librticonnextmsgcppzd.a		
	Dynamic Release	libnddscpp.so libnddsc.so libnddscore.so For *Connext Messaging*, also include: librticonnextmsgcpp.so		
	Dynamic Debug	libnddscppd.so libnddscd.so libnddscored.so For *Connext Messaging*, also include: librticonnextmsgcppd.so		
Java	Release	nddsjava.jar For *Connext Messaging*, also include: rticonnextmsg.jar	N/A	None required
	Debug	nddsjavad.jar For *Connext Messaging*, also include: rticonnextmsgd.jar		

a. The RTI C/C++ libraries are located in $(NDDSHOME)\lib\<*architecture*>\.
The RTI Java files are located in $(NDDSHOME)\class\.
(where $(NDDSHOME) is where *Connext* is installed, such as /local/rti/ndds/ndds.5.0.x)

Table 8.3 **Running Instructions for Solaris Architectures**

RTI Architecture	Library Format *(Release & Debug)*	Environment Variables
All supported Solaris architectures for Java	N/A	LD_LIBRARY_PATH= ${NDDSHOME}/lib/<*architecture*>: ${LD_LIBRARY_PATH} [a] **Note:** For all 64-bit Java architectures (...64...jdk), use **-d64** in the command line.
All supported Solaris native architectures	Static	None required
	Dynamic	LD_LIBRARY_PATH= ${NDDSHOME}/lib/<*architecture*>: ${LD_LIBRARY_PATH} [a]

a. ${NDDSHOME} represents the root directory of your *Connext* installation. ${LD_LIBRARY_PATH} represents the value of the LD_LIBRARY_PATH variable prior to changing it to support *Connext*. When using nddsjava.jar, the Java virtual machine (JVM) will attempt to load release versions of the native libraries. When using nddsjavad.jar, the JVM will attempt to load debug versions of the native libraries.

Table 8.4 **Library-Creation Details for Solaris Architectures**

RTI Architecture	Library Format	Compiler Flags Used by RTI
i86Sol2.9gcc3.3.2[a]	Static and Dynamic Release	-D_POSIX_C_SOURCE=199506L -D__EXTENSIONS__ -DSolaris2 -DSVR5 -DSUN4_SOLARIS2 -O -Wall -Wno-unknown-pragmas -fPIC -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=i386 -DTARGET=\"i86Sol2.9gcc3.3.2\" -DNDEBUG -c -Wp,-MD -Wp
	Static and Dynamic Debug	-D_POSIX_C_SOURCE=199506L -D__EXTENSIONS__ -DSolaris2 -DSVR5 -DSUN4_SOLARIS2 -g -O -Wall -Wno-unknown-pragmas -fPIC -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=i386 -DTARGET=\"i86Sol2.9gcc3.3.2\" -c -Wp,-MD -Wp
sparcSol2.9cc5.4	Static and Dynamic Release	-D_POSIX_C_SOURCE=199506L -D__EXTENSIONS__ -DSolaris2 -DSVR5 -DSUN4_SOLARIS2 -KPIC -O +w -DRTS_UNIX -DPtrIntType=long -DCPU=SPARC -DTARGET=\"sparcSol2.9cc5.4\" -DNDEBUG -c
	Static and Dynamic Debug	-D_POSIX_C_SOURCE=199506L -D__EXTENSIONS__ -DSolaris2 -DSVR5 -DSUN4_SOLARIS2 -KPIC -g +w -DRTS_UNIX -DPtrIntType=long -DCPU=SPARC -DTARGET=\"sparcSol2.9cc5.4\" -c

Table 8.4 **Library-Creation Details for Solaris Architectures**

RTI Architecture	Library Format	Compiler Flags Used by RTI
sparcSol2.10gcc3.4.2[a]	Static and Dynamic Release	-D_POSIX_C_SOURCE=199506L -D__EXTENSIONS__ -DSolaris2 -DSVR5 -DSUN4_SOLARIS2 -O -Wall -Woverloaded-virtual -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=SPARC -DTARGET=\"sparcSol2.10gcc3.4.2\" -DNDEBUG -c -Wp, -MD
	Static and Dynamic Debug	-D_POSIX_C_SOURCE=199506L -D__EXTENSIONS__ -DSolaris2 -DSVR5 -DSUN4_SOLARIS2 -g -O -Wall -Woverloaded-virtual -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=SPARC -DTARGET=\"sparcSol2.10gcc3.4.2\" -c -Wp,-MD
sparc64Sol2.10gcc3.4.2[a]	Static and Dynamic Release	-m64 -fPIC -D_POSIX_C_SOURCE=199506L -D__EXTENSIONS__ -DSolaris2 -DSVR5 -DSUN4_SOLARIS2 -O -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=SPARC -DTARGET=\"sparc64Sol2.10gcc3.4.2\" -DNDEBUG -c -Wp, -MD
	Static and Dynamic Debug	-m64 -fPIC -D_POSIX_C_SOURCE=199506L -D__EXTENSIONS__ -DSolaris2 -DSVR5 -DSUN4_SOLARIS2 -g -O -Wall -Wno-unknown-pragmas -DRTS_UNIX -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=SPARC -DTARGET=\"sparc64Sol2.10gcc3.4.2\" -c -Wp, -MD
All supported Solaris architectures for Java (*jdk)	Dynamic Release	-target 1.4 -source 1.4
	Dynamic Debug	-target 1.4 -source 1.4 -g

[a] The C++ libnddscpp dynamic libraries were linked using g++; the C dynamic libraries, i.e. libnddscore and libnddsc, were linked using gcc.

Table 8.7 **Additional Libraries for using RTI Secure WAN Transport APIs on UNIX-based Systems**

Library Format	RTI Secure WAN Transport Libraries[a]	OpenSSL Libraries[b]
Dynamic Release	libnddstransportwan.so libnddstransporttls.so	libssl.a libcrypto.a
Dynamic Debug	libnddstransportwand.so libnddstransporttlsd.so	
Static Release	libnddstransporttlsz.a libnddstransporttlszd.a	
Static Debug	libnddstransportwanz.a libnddstransportwanzd.a	

[a] The libraries are located in <*wan install dir*>/lib/<*architecture*>/. (where <*wan install dir*> is where you installed *RTI Secure WAN Transport*, such as /local/rti/ndds.5.0.x)

[b] These libraries are located <*openssl install dir*>/<*architecture*/lib, where <*openssl install dir*> is where you installed OpenSSL, such as /local/rti/openssl-0.9.8x.

9 VxWorks Platforms

Table 9.1 on page 43 lists the architectures supported on VxWorks operating systems. You can build a VxWorks application by cross-compiling from your development host.

These tables list the libraries you will need to link into your application and the required compiler flags:

- ❏ Table 9.2, “Building Instructions for VxWorks 5.x and 6.x Architectures,” on page 45
- ❏ Table 9.3, “Building Instructions for VxWorks 653 Architectures,” on page 46
- ❏ Table 9.4, “Building Instructions for VxWorks MILS Architectures,” on page 47

Compiling a *Connext* application for VxWorks depends on the development platform. For more information, such as specific compiler flags, see the *VxWorks Programmer's Guide*. Table 9.5 on page 48 provides details on how the VxWorks libraries were built. We recommend that you use similar settings.

Cross-compiling for any VxWorks platform is similar to building for a UNIX target. To build a VxWorks application, create a makefile that reflects the compiler and linker for your target with appropriate flags defined. There will be several target-specific compile flags you must set to build correctly. For more information, see the *VxWorks Programmer's Guide*.

9.1 Support for Request-Reply Communication Pattern

RTI Connext Messaging includes support for the Request-Reply Communication Pattern, for all the platforms described in Table 9.1 on page 43 and all programming languages, except as noted below.

When using C++, the following platforms do not support the Request-Reply Communication Pattern:

- ❏ ppc603Vx5.5gcc
- ❏ ppc604Vx5.5gcc

9.2 Increasing the Stack Size

Connext applications may require more than the default stack size on VxWorks.

To prevent stack overrun, you can create/enable the *DomainParticipant* in a thread with a larger stack, or increase the default stack size of the shell task by recompiling the kernel. For more information, please see the Solutions on the RTI Customer Portal, accessible from https://support.rti.com/.

9.3 Libraries for RTP on VxWorks 6

When using VxWorks 6 with Real Time Processes (RTP mode), please note the following limitations:

- ❏ Dynamic libraries are not available for VxWorks 6.3 and higher systems with RTP mode on PPC CPUs.
- ❏ Dynamic libraries for RTP mode on VxWorks 6.3 and higher systems with Pentium CPUs *are* available.

9.4 Requirement for Restarting Applications

When restarting a VxWorks application, you may need to change the 'appId' value. In general, this is only required if you still have other *Connext* applications running on other systems that were talking to the restarted application. If all the *Connext* applications are restarted, there should be no problem.

This section explains why this is necessary and how to change the appId.

All *Connext* applications must have a unique GUID (globally unique ID). This GUID is composed of a hostId and an appId. RTI implements unique appIds by using the process ID of the application. On VxWorks systems, an application's process ID will often be the same across reboots. This may cause logged errors during the discovery process, or discovery may not complete successfully for the restarted application.

The workaround is to manually provide a unique appId each time the application starts. The appId is stored in the *DomainParticipant's* WireProtocol QosPolicy. There are two general approaches to providing a unique appId. The first approach is to save the appId in NVRAM or the file system, and then increment the appId across reboots. The second approach is to base the appId on something that is likely to be different across reboots, such as a time-based register.

9.5 Multicast Support

Multicast is supported on VxWorks 5.x and 6.x; VxWorks 653; and VxWorks MILS platforms. It is configured out of the box. That is, the default value for the initial peers list (NDDS_DISCOVERY_PEERS) includes a multicast address. See the online documentation for more information.

9.6 Supported Transports

Shared memory: Shared memory is supported and enabled by default on all VxWorks 6.x architectures. It is not supported on VxWorks 5.x, VxWorks 653, or VxWorks MILS platforms.

UDPv4: Supported and enabled by default.

UDPv6: Supported on VxWorks 6.7 and higher architectures. No Traffic Class support.

TCP/IPv4: Not supported.

9.7 Monotonic Clock Support

The monotonic clock (described in Section 8.6 in the *RTI Core Libraries and Utilities User's Manual*) is supported on VxWorks 6.3 and higher architectures. This feature is not supported on VxWorks 653 2.3 or VxWorks MILS platforms.

9.8 Support for Controlling CPU Core Affinity for RTI Threads

Support for controlling CPU core affinity (described in Section 19.5 in the *RTI Core Libraries and Utilities User's Manual*) is not available for VxWorks platforms.

Table 9.1 **Supported VxWorks Target Platforms**[a]

Operating System	CPU	Compiler	RTI Architecture
VxWorks 5.5	PPC 603	gcc 2.96	ppc603Vx5.5gcc
	PPC 604	gcc 2.96	ppc604Vx5.5gcc
	PPC 750	gcc 2.96	ppc603Vx5.5gcc
	PPC 7400	gcc 2.96	ppc603Vx5.5gcc
VxWorks 6.3, 6.4	Any Wind River PPC32 CPU with floating point hardware	gcc 3.4.4	For kernel modules: ppc604Vx6.3gcc3.4.4 For Real Time Processes: ppc604Vx6.3gcc3.4.4_rtp
VxWorks 6.5	any Wind River PPC32 CPU with floating point hardware	gcc 3.4.4	For kernel modules: ppc604Vx6.5gcc3.4.4 For Real Time Processes: ppc604Vx6.5gcc3.4.4_rtp
VxWorks 6.6	Pentium	gcc 4.1.2	For Kernel Modules: pentiumVx6.6gcc4.1.2 For Real Time Processes: pentiumVx6.6gcc4.1.2_rtp
	any Wind River PPC32 CPU with floating point hardware	gcc 4.1.2	For Kernel Modules: ppc604Vx6.6gcc4.1.2 For Real Time Processes: ppc604Vx6.6gcc4.1.2_rtp
	PPC 405[b]	gcc 4.1.2	For Kernel Modules: ppc405Vx6.6gcc4.1.2 For Real Time Processes: ppc405Vx6.6gcc4.1.2_rtp
VxWorks 6.7	Pentium	gcc 4.1.2	For Kernel Modules: pentiumVx6.7gcc4.1.2 For Real Time Processes: pentiumVx6.7gcc4.1.2_rtp
	Any Wind River PPC32 CPU with floating point hardware	gcc 4.1.2	For Kernel Modules: ppc604Vx6.7gcc4.1.2 For Real Time Processes on non-SMP systems: ppc604Vx6.7gcc4.1.2_rtp For Real Time Processes on SMP systems: ppc604Vx6.7gcc4.1.2_smp
	PPC 405[b]	gcc 4.1.2	For Kernel Modules: ppc405Vx6.7gcc4.1.2 For Real Time Processes: ppc405Vx6.7gcc4.1.2_rtp

Table 9.1 **Supported VxWorks Target Platforms**[a]

Operating System	CPU	Compiler	RTI Architecture
VxWorks 6.8	Pentium	gcc 4.1.2	For Kernel Modules: pentiumVx6.8gcc4.1.2 For Real Time Processes: pentiumVx6.8gcc4.1.2_rtp
	Any Wind River PPC32 CPU with floating point hardware	gcc 4.1.2	For Kernel Modules: ppc604Vx6.8gcc4.1.2 For Real Time Processes: ppc604Vx6.8gcc4.1.2_rtp
VxWorks 6.9	Pentium32-bit	gcc 4.3.3	For Kernel Modules: pentiumVx6.9gcc4.3.3 For Real Time Processes: pentiumVx6.9gcc4.3.3_rtp
	Any Wind River PPC32 CPU with floating point hardware	gcc 4.3.3	For Kernel Modules: ppc604Vx6.9gcc4.3.3 For Real Time Processes: ppc604Vx6.9gcc4.3.3_rtp
VxWorks 653 2.3	sbc8641d	gcc 3.32	sbc8641Vx653-2.3gcc3.3.2
	SIMPC	gcc 3.32	simpcVx653-2.3gcc3.3.2
VxWorks MILS 2.1.1 with vThreads 2.2.3	ppc85xx	gcc 3.3.2	ppc85xxVxT2.2.3gcc3.3.2

[a] For use with Windows and/or Solaris Hosts as supported by Wind River Systems.

[b] For ppc405, the architecture string is the same for VxWorks 6.6 and 6.7.

Table 9.2 **Building Instructions for VxWorks 5.x and 6.x Architectures**

API	Library Format	Required RTI Libraries[a]	Required Kernel Components	Required Compiler Flags
C++	Static Release	libnddscppz.a libnddscz.a libnddscorez.a For *Connext Messaging*, also include: librticonnextmsgcppz.a	INCLUDE_TIMESTAMP For VxWorks 6.4 and below, also use: INCLUDE_ZBUF_SOCK INCLUDE_IGMP For VxWorks 6.3 and higher, also use: INCLUDE_POSIX_CLOCKS For RTI architectures with SMP support for VxWorks 6.7 and higher[b], also use: INCLUDE_TLS	-DRTI_VXWORKS
	Static Debug	libnddscppzd.a libnddsczd.a libnddscorezd.a For *Connext Messaging*, also include: librticonnextmsgcppzd.a		
	Dynamic Release	libnddscpp.so libnddsc.so libnddscore.so For *Connext Messaging*, also include: librticonnextmsgcpp.so		
	Dynamic Debug	libnddscppd.so libnddscd.so libnddscored.so For *Connext Messaging*, also include: librticonnextmsgcppd.so		
C	Static Release	libnddscz.a libnddscorez.a For *Connext Messaging*, also include: librticonnextmsgcz.a	INCLUDE_TIMESTAMP For VxWorks 6.4 and below, also use: INCLUDE_ZBUF_SOCK INCLUDE_IGMP For VxWorks 6.3 and higher, also use: INCLUDE_POSIX_CLOCKS For RTI architectures with SMP support for VxWorks 6.7 and higher[b], also use: INCLUDE_TLS	-DRTI_VXWORKS
	Static Debug	libnddsczd.a libnddscorezd.a For *Connext Messaging*, also include: librticonnextmsgczd.a		
	Dynamic Release	libnddsc.so libnddscore.so For *Connext Messaging*, also include: librticonnextmsgc.so		
	Dynamic Debug	libnddscd.so libnddscored.so For *Connext Messaging*, also include: librticonnextmsgcd.so		

[a] The *Connext* C/C++ libraries are located in $(NDDSHOME)\lib\<*architecture*>\.
(where $(NDDSHOME) is where *Connext* is installed, such as c:\rti\ndds.5.0.x)

[b] In this version, only ppc604Vx6.7gcc4.1.2_smp

Table 9.3 **Building Instructions for VxWorks 653 Architectures**

API	Library Format	Required RTI Libraries[a]	Required Kernel Components	Required Compiler Flags
C++	Static Release	libnddscppz.a libnddscz.a libnddscorez.a For *Connext Messaging*, also include: librticonnextmsgcppz.a	See either: Table 9.6, "Required Kernel Components for sbc8641Vx653-2.3gcc3.3.2," on page 54 or Table 9.7, "Required Kernel Components for simpcVx653-2.3gcc3.3.2," on page 55	-DRTI_VXWORKS -DRTI_VX653
	Static Debug	libnddscppzd.a libnddsczd.a libnddscorezd.a For *Connext Messaging*, also include: librticonnextmsgcppzd.a		
	Dynamic Release	libnddscpp.so libnddsc.so libnddscore.so For *Connext Messaging*, also include: librticonnextmsgcpp.so		
	Dynamic Debug	libnddscppd.so libnddscd.so libnddscored.so For *Connext Messaging*, also include: librticonnextmsgcppd.so		
C	Static Release	libnddscz.a libnddscorez.a For *Connext Messaging*, also include: librticonnextmsgcz.a		
	Static Debug	libnddsczd.a libnddscorezd.a For *Connext Messaging*, also include: librticonnextmsgczd.a		
	Dynamic Release	libnddsc.so libnddscore.so For *Connext Messaging*, also include: librticonnextmsgc.so		
	Dynamic Debug	libnddscd.so libnddscored.so For *Connext Messaging*, also include: librticonnextmsgcd.so		

[a] The *Connext* C/C++ libraries are located in $(NDDSHOME)\lib\<*architecture*>\.
(where $(NDDSHOME) is where *Connext* is installed, such as c:\rti\ndds.5.0.x)

Table 9.4 **Building Instructions for VxWorks MILS Architectures**

API	Library Format	Required RTI Libraries[a]	Required Kernel Components	Required Compiler Flags
C++	Static Release	libnddscppz.a libnddscz.a libnddscorez.a For *Connext Messaging*, also include: librticonnextmsgcppz.a	The MILS 2.1.1 patch that corrects defect number WIND00343321 must be installed for *Connext* libraries to work on a MILS 2.1.1 system. This patch can be obtained through the regular Wind River support channel.	-DRTI_VXWORKS
	Static Debug	libnddscppzd.a libnddsczd.a libnddscorezd.a For *Connext Messaging*, also include: librticonnextmsgcppzd.a		
	Dynamic Release	libnddscpp.so libnddsc.so libnddscore.so For *Connext Messaging*, also include: librticonnextmsgcpp.so		
	Dynamic Debug	libnddscppd.so libnddscd.so libnddscored.so For *Connext Messaging*, also include: librticonnextmsgcppd.so		
C	Static Release	libnddscz.a libnddscorez.a For *Connext Messaging*, also include: librticonnextmsgcz.a		
	Static Debug	libnddsczd.a libnddscorezd.a For *Connext Messaging*, also include: librticonnextmsgczd.a		
	Dynamic Release	libnddsc.so libnddscore.so For *Connext Messaging*, also include: librticonnextmsgc.so		
	Dynamic Debug	libnddscd.so libnddscored.so For *Connext Messaging*, also include: librticonnextmsgcd.so		

[a] The *Connext* C/C++ libraries are located in $(NDDSHOME)\lib\<architecture>\.
(where $(NDDSHOME) is where *Connext* is installed, such as c:\rti\ndds.5.0.x)

Table 9.5 **Library-Creation Details for All VxWorks Architectures**

RTI Architecture	Library Format	Compiler Flags Used by RTI
pentiumVx6.6gcc4.1.2	Static or Dynamic Release	-march=pentium -fno-builtin -ansi -DTOOL=gnu -D_WRS_KERNEL -D_PROTOTYPE_5_0 -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=6 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DTOOL_FAMILY=gnu -DPtrIntType=long -DCPU=PENTIUM -DNDEBUG -c -Wp,-MD
	Static or Dynamic Debug	-march=pentium -fno-builtin -ansi -DTOOL=gnu -D_WRS_KERNEL -D_PROTOTYPE_5_0 -g -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=6 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DTOOL_FAMILY=gnu -DPtrIntType=long -DCPU=PENTIUM -c -Wp,-MD
pentiumVx6.6gcc4.1.2_rtp	Static Release	-march=i486 -ansi -DTOOL=gnu -mrtp -D_PROTOTYPE_5_0 -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=6 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DTOOL_FAMILY=gnu -DPtrIntType=long -DCPU=PENTIUM -DNDEBUG -c -Wp,-MD
	Static Debug	-march=i486 -ansi -DTOOL=gnu -mrtp -D_PROTOTYPE_5_0 -g -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=6 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DTOOL_FAMILY=gnu -DPtrIntType=long -DCPU=PENTIUM -c -Wp,-MD
	Dynamic Release	-march=i486 -ansi -DTOOL=gnu -mrtp -D_PROTOTYPE_5_0 -fPIC -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=6 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DTOOL_FAMILY=gnu -DPtrIntType=long -DCPU=PENTIUM -DNDEBUG -c -Wp,-MD
	Dynamic Debug	-march=i486 -ansi -DTOOL=gnu -mrtp -D_PROTOTYPE_5_0 -fPIC -g -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=6 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DTOOL_FAMILY=gnu -DPtrIntType=long -DCPU=PENTIUM -c -Wp,-MD
pentiumVx6.7gcc4.1.2	Static or Dynamic Release	-march=pentium -fno-builtin -ansi -DTOOL=gnu -D_WRS_KERNEL -D_PROTOTYPE_5_0 -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=7 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DTOOL_FAMILY=gnu -DPtrIntType=long -DCPU=PENTIUM -DNDEBUG -c -Wp,-MD
	Static or Dynamic Debug	-march=pentium -fno-builtin -ansi -DTOOL=gnu -D_WRS_KERNEL -D_PROTOTYPE_5_0 -g -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=7 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DTOOL_FAMILY=gnu -DPtrIntType=long -DCPU=PENTIUM -c -Wp,-MD
pentiumVx6.7gcc4.1.2_rtp	Static Release	-march=i486 -ansi -DTOOL=gnu -mrtp -D_PROTOTYPE_5_0 -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=7 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DTOOL_FAMILY=gnu -DPtrIntType=long -DCPU=PENTIUM -DNDEBUG -c -Wp,-MD
	Static Debug	-march=i486 -ansi -DTOOL=gnu -mrtp -D_PROTOTYPE_5_0 -g -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=7 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DTOOL_FAMILY=gnu -DPtrIntType=long -DCPU=PENTIUM -c -Wp,-MD
	Dynamic Release	-march=i486 -ansi -DTOOL=gnu -mrtp -D_PROTOTYPE_5_0 -fPIC -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=7 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DTOOL_FAMILY=gnu -DPtrIntType=long -DCPU=PENTIUM -DNDEBUG -c -Wp,-MD
	Dynamic Debug	-march=i486 -ansi -DTOOL=gnu -mrtp -D_PROTOTYPE_5_0 -fPIC -g -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=7 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DTOOL_FAMILY=gnu -DPtrIntType=long -DCPU=PENTIUM -c -Wp,-MD

Table 9.5 **Library-Creation Details for All VxWorks Architectures**

RTI Architecture	Library Format	Compiler Flags Used by RTI
pentiumVx6.8gcc4.1.2	Static or Dynamic Release	ccpentium -m32 -march=pentium -fno-builtin -ansi -DCPU=PENTIUM -DTOOL_FAMILY=gnu -DTOOL=gnu -D_WRS_KERNEL -D__PROTOTYPE_5_0 -O -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=8 -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DPtrIntType=long -DCSREAL_IS_FLOAT -DNDEBUG -Wp,-MD
	Static or Dynamic Debug	ccpentium -m32 -march=pentium -fno-builtin -ansi -DCPU=PENTIUM -DTOOL_FAMILY=gnu -DTOOL=gnu -D_WRS_KERNEL -D__PROTOTYPE_5_0 -g -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=8 -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DPtrIntType=long -DCSREAL_IS_FLOAT -Wp,-MD
pentiumVx6.8gcc4.1.2_rtp	Static or Dynamic Release	ccpentium -m32 -march=pentium -ansi -DCPU=PENTIUM -DTOOL_FAMILY=gnu -DTOOL=gnu -mrtp -D__PROTOTYPE_5_0 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=8 -DPtrIntType=long -DCSREAL_IS_FLOAT -DNDEBUG -Wp,-MD
	Static or Dynamic Debug	ccpentium -m32 -march=pentium -ansi -DCPU=PENTIUM -DTOOL_FAMILY=gnu -DTOOL=gnu -mrtp -D__PROTOTYPE_5_0 -g -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=8 -DPtrIntType=long -DCSREAL_IS_FLOAT -Wp,-MD
pentiumVx6.9gcc4.3.3	Static or Dynamic Release	ccpentium -m32 -march=pentium -fno-builtin -ansi -DCPU=PENTIUM -DTOOL_FAMILY=gnu -DTOOL=gnu -D_WRS_KERNEL -D__PROTOTYPE_5_0 -O -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=9 -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DPtrIntType=long -DCSREAL_IS_FLOAT -DNDEBUG -Wp,-MD
	Static or Dynamic Debug	ccpentium -m32 -march=pentium -fno-builtin -ansi -DCPU=PENTIUM -DTOOL_FAMILY=gnu -DTOOL=gnu -D_WRS_KERNEL -D__PROTOTYPE_5_0 -g -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=9 -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=PENTIUM -Wp,-MD
pentiumVx6.9gcc4.3.3_rtp	Static or Dynamic Release	ccpentium -m32 -march=pentium -ansi -DCPU=PENTIUM -DTOOL_FAMILY=gnu -DTOOL=gnu -mrtp -D__PROTOTYPE_5_0 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=9 -DPtrIntType=long -DCSREAL_IS_FLOAT -DNDEBUG -Wp,-MD
	Static or Dynamic Debug	ccpentium -m32 -march=pentium -ansi -DCPU=PENTIUM -DTOOL_FAMILY=gnu -DTOOL=gnu -mrtp -D__PROTOTYPE_5_0 -g -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=9 -DPtrIntType=long -DCSREAL_IS_FLOAT -Wp,-MD
ppc405Vx6.6gcc4.1.2	Static or Dynamic Release	-mcpu=405 -fno-builtin -mlongcall -DTOOL=gnu -mstrict-align -msoft-float -ansi -D_WRS_KERNEL -D_PROTOTYPE_5_0 -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=6 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DTOOL=gnu -DTOOL_FAMILY=gnu -DPtrIntType=long -DCPU=PPC405 -DNDEBUG -c -Wp,-MD
	Static or Dynamic Debug	-mcpu=405 -fno-builtin -mlongcall -DTOOL=gnu -mstrict-align -msoft-float -ansi -D_WRS_KERNEL -D_PROTOTYPE_5_0 -g -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=6 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DTOOL=gnu -DTOOL_FAMILY=gnu -DPtrIntType=long -DCPU=PPC405 -c -Wp,-MD

Table 9.5 **Library-Creation Details for All VxWorks Architectures**

RTI Architecture	Library Format	Compiler Flags Used by RTI
ppc405Vx6.6gcc4.1.2_rtp	Static Release	-msoft-float -mlongcall -mregnames -mstrict-align -ansi -DTOOL=gnu -mrtp -D_PROTOTYPE_5_0 -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=6 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DTOOL=sfgnu -DTOOL_FAMILY=gnu -DPtrIntType=long -DCPU=PPC32 -DNDEBUG -c -Wp,-MD
	Static Debug	-msoft-float -mlongcall -mregnames -mstrict-align -ansi -DTOOL=gnu -mrtp -fPIC -shared -D_PROTOTYPE_5_0 -g -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=6 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DTOOL=sfgnu -DTOOL_FAMILY=gnu -DPtrIntType=long -DCPU=PPC32 -c -Wp,-MD
ppc603Vx5.5gcc	Static or Dynamic Release	-mcpu=603 -G 0 -fno-builtin -mlongcall -D__PROTOTYPE_5_0 -DVXWORKS_MAJOR_VERSION=5 -DVXWORKS_MINOR_VERSION=5 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=PPC603 -DNDEBUG -c -Wp,-MD
	Static or Dynamic Debug	-mcpu=603 -G 0 -fno-builtin -mlongcall -D__PROTOTYPE_5_0 -g -DVXWORKS_MAJOR_VERSION=5 -DVXWORKS_MINOR_VERSION=5 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=PPC603 -c -Wp,-MD
ppc604Vx5.5gcc	Static or Dynamic Release	-mcpu=604 -G 0 -fno-builtin -mlongcall -D__PROTOTYPE_5_0 -DVXWORKS_MAJOR_VERSION=5 -DVXWORKS_MINOR_VERSION=5 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DPtrIntType=long -DCPU=PPC604 -DNDEBUG -c -Wp,-MD
	Static or Dynamic Debug	-mcpu=604 -G 0 -fno-builtin -mlongcall -D__PROTOTYPE_5_0 -g -DVXWORKS_MAJOR_VERSION=5 -DVXWORKS_MINOR_VERSION=5 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DPtrIntType=long -DCPU=PPC604 -c -Wp,-MD
ppc604Vx6.3gcc3.4.4	Static or Dynamic Release	-mcpu=604 -fno-builtin -mlongcall -DTOOL=gnu -mstrict-align -mno-implicit-fp -ansi -D_WRS_KERNEL -D_PROTOTYPE_5_0 -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=3 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DTOOL_FAMILY=gnu -DPtrIntType=long -DCPU=PPC604 -DNDEBUG -c -Wp,-MD
	Static or Dynamic Debug	-mcpu=604 -fno-builtin -mlongcall -DTOOL=gnu -mstrict-align -mno-implicit-fp -ansi -D_WRS_KERNEL -D_PROTOTYPE_5_0 -g -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=3 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DTOOL_FAMILY=gnu -DPtrIntType=long -DCPU=PPC604 -c -Wp,-MD
ppc604Vx6.3gcc3.4.4_rtp	Static Release	-mhard-float -mlongcall -mregnames -mstrict-align -ansi -DTOOL=gnu -mrtp -D_PROTOTYPE_5_0 -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=3 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DTOOL_FAMILY=gnu -DPtrIntType=long -DCPU=PPC32 -DNDEBUG -c -Wp,-MD
	Static Debug	-mhard-float -mlongcall -mregnames -mstrict-align -ansi -DTOOL=gnu -mrtp -fPIC -shared -D_PROTOTYPE_5_0 -g -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=3 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DTOOL_FAMILY=gnu -DPtrIntType=long -DCPU=PPC32 -c -Wp,-MD

Table 9.5 **Library-Creation Details for All VxWorks Architectures**

RTI Architecture	Library Format	Compiler Flags Used by RTI
ppc604Vx6.5gcc3.4.4	Static or Dynamic Release	-mcpu=604 -mstrict-align -fno-builtin -ansi -mlongcall -mno-implicit-fp -D_WRS_KERNEL -D_PROTOTYPE_5_0 -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=5 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DTOOL=gnu -DTOOL_FAMILY=gnu -DPtrIntType=long -DCPU=PPC604 -DNDEBUG -c -Wp,-MD
	Static or Dynamic Debug	-mcpu=604 -mstrict-align -fno-builtin -ansi -mlongcall -mno-implicit-fp -D_WRS_KERNEL -D_PROTOTYPE_5_0 -g -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=5 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DTOOL=gnu -DTOOL_FAMILY=gnu -DPtrIntType=long -DCPU=PPC604 -c -Wp,-MD
ppc604Vx6.5gcc3.4.4_rtp	Static Release	-mhard-float -mstrict-align -ansi -mregnames -mlongcall -mrtp -D_PROTOTYPE_5_0 -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=5 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DTOOL=gnu -DTOOL_FAMILY=gnu -DPtrIntType=long -DCPU=PPC32 -DNDEBUG -c -Wp,-MD
	Static Debug	-mhard-float -mstrict-align -ansi -mregnames -mlongcall -mrtp -D_PROTOTYPE_5_0 -g -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=5 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DTOOL=gnu -DTOOL_FAMILY=gnu -DPtrIntType=long -DCPU=PPC32 -c -Wp,-MD
ppc604Vx6.6gcc4.1.2	Static or Dynamic Release	-mcpu=604 -fno-builtin -mlongcall -DTOOL=gnu -mstrict-align -ansi -D_WRS_KERNEL -D_PROTOTYPE_5_0 -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=6 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DTOOL=gnu -DTOOL_FAMILY=gnu -DPtrIntType=long -DCPU=PPC604 -DNDEBUG -c -Wp,-MD
	Static or Dynamic Debug	-mcpu=604 -fno-builtin -mlongcall -DTOOL=gnu -mstrict-align -ansi -D_WRS_KERNEL -D_PROTOTYPE_5_0 -g -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=6 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DTOOL=gnu -DTOOL_FAMILY=gnu -DPtrIntType=long -DCPU=PPC604 -c -Wp,-MD
ppc604Vx6.6gcc4.1.2_rtp	Static Release	-mhard-float -mlongcall -mregnames -mstrict-align -ansi -mrtp -D_PROTOTYPE_5_0 -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=6 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DTOOL=gnu -DTOOL_FAMILY=gnu -DPtrIntType=long -DCPU=PPC32 -DNDEBUG -c -Wp,-MD
	Static Debug	-mhard-float -mlongcall -mregnames -mstrict-align -ansi -mrtp -fPIC -shared -D_PROTOTYPE_5_0 -g -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=6 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DTOOL=gnu -DTOOL_FAMILY=gnu -DPtrIntType=long -DCPU=PPC32 -c -Wp,-MD
ppc604Vx6.7gcc4.1.2	Static or Dynamic Release	-mcpu=604 -fno-builtin -mlongcall -DTOOL=gnu -mstrict-align -ansi -D_WRS_KERNEL -D_PROTOTYPE_5_0 -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=7 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DTOOL=gnu -DTOOL_FAMILY=gnu -DPtrIntType=long -DCPU=PPC604 -DNDEBUG -c -Wp,-MD
	Static or Dynamic Debug	-mcpu=604 -fno-builtin -mlongcall -DTOOL=gnu -mstrict-align -ansi -D_WRS_KERNEL -D_PROTOTYPE_5_0 -g -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=7 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DTOOL=gnu -DTOOL_FAMILY=gnu -DPtrIntType=long -DCPU=PPC604 -c -Wp,-MD

Table 9.5 **Library-Creation Details for All VxWorks Architectures**

RTI Architecture	Library Format	Compiler Flags Used by RTI
ppc604Vx6.7gcc4.1.2_rtp, ppc604Vx6.7gcc4.1.2_smp	Static Release	-mhard-float -mlongcall -mregnames -mstrict-align -ansi -mrtp -D_PROTOTYPE_5_0 -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=7 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DTOOL=gnu -DTOOL_FAMILY=gnu -DPtrIntType=long -DCPU=PPC32 -DNDEBUG -c -Wp,-MD
	Static Debug	-mhard-float -mlongcall -mregnames -mstrict-align -ansi -mrtp -fPIC -shared -D_PROTOTYPE_5_0 -g -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=7 -O -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DTOOL=gnu -DTOOL_FAMILY=gnu -DPtrIntType=long -DCPU=PPC32 -c -Wp,-MD
ppc604Vx6.8gcc4.1.2	Static or Dynamic Release	ccppc -m32 -mstrict-align -ansi -fno-builtin -mlongcall -DCPU=PPC32 -DTOOL_FAMILY=gnu -DTOOL=gnu -D_WRS_KERNEL -D__PROTOTYPE_5_0 -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=8 -O2 -fno-strict-aliasing -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DPtrIntType=long -DCSREAL_IS_FLOAT -DNDEBUG -Wp,-MD
	Static or Dynamic Debug	ccppc -m32 -mstrict-align -ansi -fno-builtin -mlongcall -DCPU=PPC32 -DTOOL_FAMILY=gnu -DTOOL=gnu -D_WRS_KERNEL -D__PROTOTYPE_5_0 -g -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=8 -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DPtrIntType=long -DCSREAL_IS_FLOAT -Wp,-MD
ppc604Vx6.8gcc4.1.2_rtp	Static or Dynamic Release	ccppc -m32 -mhard-float -mstrict-align -mregnames -ansi -mlongcall -DCPU=PPC32 -DTOOL_FAMILY=gnu -DTOOL=gnu -mrtp -D__PROTOTYPE_5_0 -O2 -fno-strict-aliasing -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=8 -DPtrIntType=long -DCSREAL_IS_FLOAT -DNDEBUG -Wp,-MD
	Static or Dynamic Debug	ccppc -m32 -mhard-float -mstrict-align -mregnames -ansi -mlongcall -DCPU=PPC32 -DTOOL_FAMILY=gnu -DTOOL=gnu -mrtp -D__PROTOTYPE_5_0 -g -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=8 -DPtrIntType=long -DCSREAL_IS_FLOAT -Wp,-MD
ppc604Vx6.9gcc4.3.3	Static Release	ccppc -m32 -mstrict-align -ansi -fno-builtin -mlongcall -DCPU=PPC32 -DTOOL_FAMILY=gnu -DTOOL=gnu -D_WRS_KERNEL -D__PROTOTYPE_5_0 -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=9 -O2 -fno-strict-aliasing -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DPtrIntType=long -DCSREAL_IS_FLOAT -DNDEBUG -Wp,-MD
	Static Debug	ccppc -m32 -mstrict-align -ansi -fno-builtin -mlongcall -DCPU=PPC32 -DTOOL_FAMILY=gnu -DTOOL=gnu -D_WRS_KERNEL -D__PROTOTYPE_5_0 -g -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=9 -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DPtrIntType=long -DCSREAL_IS_FLOAT -Wp,-MD
ppc604Vx6.9gcc4.3.3_rtp	Static Release	ccppc -mhard-float -mstrict-align -m32 -mregnames -ansi -mlongcall -DCPU=PPC32 -DTOOL_FAMILY=gnu -DTOOL=gnu -mrtp -D__PROTOTYPE_5_0 -O2 -fno-strict-aliasing -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=9 -DPtrIntType=long -DCSREAL_IS_FLOAT -DNDEBUG -Wp,-MD
	Static Debug	ccppc -mhard-float -mstrict-align -m32 -mregnames -ansi -mlongcall -DCPU=PPC32 -DTOOL_FAMILY=gnu -DTOOL=gnu -mrtp -D__PROTOTYPE_5_0 -g -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DVXWORKS_MAJOR_VERSION=6 -DVXWORKS_MINOR_VERSION=9 -DPtrIntType=long -DCSREAL_IS_FLOAT -Wp,-MD

Table 9.5 **Library-Creation Details for All VxWorks Architectures**

RTI Architecture	Library Format	Compiler Flags Used by RTI
ppc85xxVxT2.2.3gcc3.3.2	Static or Dynamic Release	-fno-zero-initialized-in-bss -mcpu=8540 -mvthreads -mlongcall -mstrict-align -mabi=no-spe -msoft-float -G 0 -fvolatile -fno-builtin -O -Wall -Wno-unknown-pragmas -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=PPC85XX -DNDEBUG
	Static or Dynamic Debug	-fno-zero-initialized-in-bss -mcpu=8540 -mvthreads -mlongcall -mstrict-align -mabi=no-spe -msoft-float -G 0 -fvolatile -fno-builtin -g -Wall -Wno-unknown-pragmas -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=PPC85XX
sbc8641Vx653-2.3gcc3.3.2	Static or Dynamic Release	-DTOOL_FAMILY=gnu -DTOOL=gnu -mlongcall -Wall -G 0 -fno-builtin -mlongcall -D_WRS_KERNEL -D__PROTOTYPE_5_0 -DVXWORKS_MAJOR_VERSION=5 -DVXWORKS_MINOR_VERSION=5 -O -Wno-unknown-pragmas -DRTS_VXWORKS -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=PPC604 -DNDEBUG -c -Wp,-MD
	Static or Dynamic Debug	-DTOOL_FAMILY=gnu -DTOOL=gnu -mlongcall -Wall -G 0 -fno-builtin -mlongcall -D_WRS_KERNEL -D__PROTOTYPE_5_0 -g -DVXWORKS_MAJOR_VERSION=5 -DVXWORKS_MINOR_VERSION=5 -Wall -Wno-unknown-pragmas -DRTS_VXWORKS -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=PPC604 -c -Wp,-MD
simpcVx653-2.3gcc3.3.2	Static or Dynamic Release	-DTOOL_FAMILY=gnu -DTOOL=gnu -DCPU=SIMNT -Wall -nostdlib -fno-defer-pop -fno-builtin -mcpu=pentium -D_WRS_KERNEL -D__PROTOTYPE_5_0 -DVXWORKS_MAJOR_VERSION=5 -DVXWORKS_MINOR_VERSION=5 -O -Wno-unknown-pragmas -DRTS_VXWORKS -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=SIMNT -DNDEBUG -c -Wp,-MD
	Static or Dynamic Debug	-DTOOL_FAMILY=gnu -DTOOL=gnu -DCPU=SIMNT -Wall -nostdlib -fno-defer-pop -fno-builtin -mcpu=pentium -D_WRS_KERNEL -D__PROTOTYPE_5_0 -g -DVXWORKS_MAJOR_VERSION=5 -DVXWORKS_MINOR_VERSION=5 -Wno-unknown-pragmas -DRTS_VXWORKS -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=SIMNT -c -Wp,-MD

Table 9.6 **Required Kernel Components for sbc8641Vx653-2.3gcc3.3.2**[a]

INCLUDE_ARINC_SCHEDULER_INIT	INCLUDE_NETINET_IF_SUBR
INCLUDE_ARP_API	INCLUDE_NETINET_IGMP
INCLUDE_AUXCLK	INCLUDE_NETINET_IN
INCLUDE_BOOT_LINE	INCLUDE_NETINET_IN_CKSUM
INCLUDE_BOOT_LINE_INIT	INCLUDE_NETINET_IN_PCB
INCLUDE_BSD_SOCKET	INCLUDE_NETINET_IN_PROTO
INCLUDE_BSP_MODULES	INCLUDE_NETINET_IP_ICMP
INCLUDE_BSP_VXWORKS	INCLUDE_NETINET_IP_INPUT
INCLUDE_BYTENVRAM	INCLUDE_NETINET_IP_OUTPUT
INCLUDE_DEBUG_CORE	INCLUDE_NETINET_RADIX
INCLUDE_DEBUG_UTIL	INCLUDE_NETINET_RAW_IP
INCLUDE_END	INCLUDE_NETINET_ROUTE
INCLUDE_END_BOOT	INCLUDE_NETINET_SYS_SOCKET
INCLUDE_EXC_SHOW_INIT	INCLUDE_NETINET_UDP_USRREQ
INCLUDE_FLASHMEM	INCLUDE_NETINET_UIPC_DOM
INCLUDE_FTP	INCLUDE_NETINET_UIPC_MBUF
INCLUDE_HOST_TBL	INCLUDE_NETINET_UIPC_SOCK
INCLUDE_ICMP	INCLUDE_NETINET_UIPC_SOCK2
INCLUDE_IGMP	INCLUDE_NETINET_UNIXLIB
INCLUDE_IO_EXTRA_INIT	INCLUDE_NETMASK_GET
INCLUDE_IO_SYSTEM_INIT	INCLUDE_NETWORK
INCLUDE_IP	INCLUDE_NETWRS_ETHERMULTILIB
INCLUDE_KERNEL_BASIC	INCLUDE_NETWRS_IFLIB
INCLUDE_KERNEL_BASIC_INIT	INCLUDE_NETWRS_INETLIB
INCLUDE_KERNEL_BASIC_INIT2	INCLUDE_NETWRS_NETBUFLIB
INCLUDE_KERNEL_CORE	INCLUDE_NETWRS_REMLIB
INCLUDE_KERNEL_FULL	INCLUDE_NETWRS_ROUTELIB
INCLUDE_KERNEL_NORMAL_MODE	INCLUDE_NETWRS_XDR
INCLUDE_KERNEL_SHOW	INCLUDE_NV_RAM
INCLUDE_KERNEL_UTIL	INCLUDE_PARTITION_INIT
INCLUDE_LOADER	INCLUDE_POST_KERNEL_CORE_INIT
INCLUDE_LOADER_EXTRA	INCLUDE_POST_KERNEL_CORE_INIT2
INCLUDE_LOOPBACK	INCLUDE_PPCDECTIMER
INCLUDE_MIILIB	INCLUDE_PRE_KERNEL_CORE_INIT
INCLUDE_MMU_BASIC	INCLUDE_SERIAL
INCLUDE_MOTTSECEND	INCLUDE_SHELL
INCLUDE_MUX	INCLUDE_SHELL_VI_MODE
INCLUDE_NET_DRV	INCLUDE_SOCKET_DEV
INCLUDE_NET_HOST_SETUP	INCLUDE_SYM_TBL_INIT
INCLUDE_NET_INIT	INCLUDE_SYSCLK
INCLUDE_NET_LIB	INCLUDE_SYSTEM_START_INIT
INCLUDE_NET_RANDOM	INCLUDE_TCP

Table 9.6 **Required Kernel Components for sbc8641Vx653-2.3gcc3.3.2**[a]

INCLUDE_NET_REM_IO	INCLUDE_TFTP_CLIENT
INCLUDE_NET_SETUP	INCLUDE_TIME_MONITOR_INIT
INCLUDE_NET_SYM_TBL	INCLUDE_UDP
INCLUDE_NET_TASK	INCLUDE_USER_APPL
INCLUDE_NETDEV_CONFIG	INCLUDE_USR_DEVSPLIT
INCLUDE_NETDEV_NAMEGET	INCLUDE_USR_FS_UTILS
INCLUDE_NETINET_IF	INCLUDE_WDB
INCLUDE_NETINET_IF_ETHER	INCLUDE_WDB_COMM_END

a. Install partition_socket_driver_v1.3. Follow instructions from Wind River for the installation.

Table 9.7 **Required Kernel Components for simpcVx653-2.3gcc3.3.2**[a]

INCLUDE_ARINC_SCHEDULER_INIT	INCLUDE_NETINET_IN_PCB
INCLUDE_ARP_API	INCLUDE_NETINET_IN_PROTO
INCLUDE_BOOT_LINE	INCLUDE_NETINET_IP_ICMP
INCLUDE_BOOT_LINE_INIT	INCLUDE_NETINET_IP_INPUT
INCLUDE_BSD_SOCKET	INCLUDE_NETINET_IP_OUTPUT
INCLUDE_BSP_MODULES	INCLUDE_NETINET_RADIX
INCLUDE_BSP_VXWORKS	INCLUDE_NETINET_RAW_IP
INCLUDE_DEBUG_CORE	INCLUDE_NETINET_ROUTE
INCLUDE_DEBUG_UTIL	INCLUDE_NETINET_SYS_SOCKET
INCLUDE_END	INCLUDE_NETINET_UDP_USRREQ
INCLUDE_END_BOOT	INCLUDE_NETINET_UIPC_DOM
INCLUDE_FTP	INCLUDE_NETINET_UIPC_MBUF
INCLUDE_HOST_TBL	INCLUDE_NETINET_UIPC_SOCK
INCLUDE_ICMP	INCLUDE_NETINET_UIPC_SOCK2
INCLUDE_IGMP	INCLUDE_NETINET_UNIXLIB
INCLUDE_IO_EXTRA_INIT	INCLUDE_NETMASK_GET
INCLUDE_IO_SYSTEM_INIT	INCLUDE_NETWORK
INCLUDE_IP	INCLUDE_NETWRS_ETHERMULTILIB
INCLUDE_KERNEL_BASIC	INCLUDE_NETWRS_IFLIB
INCLUDE_KERNEL_BASIC_INIT	INCLUDE_NETWRS_INETLIB
INCLUDE_KERNEL_BASIC_INIT2	INCLUDE_NETWRS_NETBUFLIB
INCLUDE_KERNEL_CORE	INCLUDE_NETWRS_REMLIB
INCLUDE_KERNEL_FULL	INCLUDE_NETWRS_ROUTELIB
INCLUDE_KERNEL_NORMAL_MODE	INCLUDE_NETWRS_XDR
INCLUDE_LOOPBACK	INCLUDE_NTEND
INCLUDE_MUX	INCLUDE_NTPASSFS
INCLUDE_NET_DRV	INCLUDE_NULLNVRAM
INCLUDE_NET_HOST_SETUP	INCLUDE_PARTITION_INIT
INCLUDE_NET_INIT	INCLUDE_POST_KERNEL_CORE_INIT
INCLUDE_NET_LIB	INCLUDE_POST_KERNEL_CORE_INIT2
INCLUDE_NET_RANDOM	INCLUDE_PRE_KERNEL_CORE_INIT

Table 9.7 **Required Kernel Components for simpcVx653-2.3gcc3.3.2**[a]

INCLUDE_NET_REM_IO	INCLUDE_SIMPCTIMER
INCLUDE_NET_SETUP	INCLUDE_SOCKET_DEV
INCLUDE_NET_TASK	INCLUDE_SYSTEM_START_INIT
INCLUDE_NETDEV_CONFIG	INCLUDE_TCP
INCLUDE_NETDEV_NAMEGET	INCLUDE_TFTP_CLIENT
INCLUDE_NETINET_IF	INCLUDE_TIME_MONITOR_INIT
INCLUDE_NETINET_IF_ETHER	INCLUDE_UDP
INCLUDE_NETINET_IF_SUBR	INCLUDE_USER_APPL
INCLUDE_NETINET_IGMP	INCLUDE_WDB
INCLUDE_NETINET_IN	INCLUDE_WDB_COMM_END
INCLUDE_NETINET_IN_CKSUM	INCLUDE_WINSIO

[a] Install partition_socket_driver_v1.3. Follow instructions from Wind River for the installation.

10 Windows Platforms

First, see the basic instructions for compiling on Microsoft® Windows® systems provided in Section 9.4 in the *RTI Core Libraries and Utilities User's Manual*. The following tables provide supplemental information. Table 10.1 lists the architectures supported on Windows operating systems.

Table 10.1 **Supported Windows Architectures**

Operating System	CPU	Compiler or Software Development Kit [a] [b]	RTI Architecture
Windows 7 32-bit Edition	x86	Visual Studio 2010	i86Win32VS2010
		Visual Studio 2010 (C++/CLI, C# 8.0 or 9.0)	i86Win32dotnet4.0
		Sun Java Platform Standard Edition JDK 1.6	i86Win32jdk
Windows 7 64-bit Edition	x64	Visual Studio 2010	x64Win64VS2010
		Visual Studio 2010 (C++/CLI, C# 8.0 or 9.0)	x64Win64dotnet4.0
		Sun Java Platform Standard Edition JDK 1.6	x64Win64jdk
Windows 2003 32-bit Edition	x86	Visual Studio 2005 SP 1	i86Win32VS2005
		Visual Studio 2005 SP 1 (C++/CLI, C# 8.0 or 9.0)	i86Win32dotnet2.0
		Visual Studio 2008 SP1	i86Win32VS2008
		Visual Studio 2008 SP 1 (C++/CLI, C# 8.0 or 9.0)	i86Win32dotnet2.0
		Sun Java Platform Standard Edition JDK 1.6	i86Win32jdk
Windows 2003 64-bit Edition	x64	Visual Studio 2005 SP 1	x64Win64VS2005
		Visual Studio 2005 SP 1 (C++/CLI, C# 8.0 or 9.0)	x64Win64dotnet2.0
		Visual Studio 2008 SP 1	x64Win64VS2008
		Sun Java Platform Standard Edition JDK 1.6	x64Win64jdk
Windows Server 2008 R2 64-bit Edition	x64	Visual Studio 2005 SP 1 (C++, C# 8.0 or 9.0)	x64Win64dotnet2.0
		Visual Studio 2010	x64Win64VS2010
		Visual Studio 2010 (C++/CLI, C# 8.0 or 9.0)	x64Win64dotnet4.0
		Sun Java Platform Standard Edition JDK 1.6	x64Win64jdk
Windows Vista 32-bit Edition	x86	Visual Studio 2005 SP 1	i86Win32VS2005
		Visual Studio 2005 SP 1 (C++/CLI, C# 8.0 or 9.0)	i86Win32dotnet2.0
		Visual Studio 2008 SP 1	i86Win32VS2008
		Sun Java Platform Standard Edition JDK 1.6	i86Win32jdk
Windows Vista 64-bit Edition	x64	Visual Studio 2005 SP 1	x64Win64VS2005
		Visual Studio 2005 SP 1 (C++/CLI, C# 8.0 or 9.0)	x64Win64dotnet2.0
		Visual Studio 2008 SP1	x64Win64VS2008
		Sun Java Platform Standard Edition JDK 1.6	x64Win64jdk

Table 10.1 **Supported Windows Architectures**

Operating System	CPU	Compiler or Software Development Kit [a] [b]	RTI Architecture
Windows XP[c] 32-bit Professional Edition Service Pack 2	x86	Visual Studio 2005 SP 1	i86Win32VS2005
		Visual Studio 2005 SP 1 (C++/CLI, C# 8.0 or 9.0)	i86Win32dotnet2.0
		Visual Studio 2008 SP 1	i86Win32VS2008
		Sun Java Platform Standard Edition JDK .6	i86Win32jdk
Windows XP 64-bit Professional Edition Service Pack 2	x64	Visual Studio 2005 SP 1	x64Win64VS2005
		Visual Studio 2005 SP 1 (C++/CLI, C# 8.0 or 9.0)	x64Win64dotnet2.0
		Visual Studio 2008 SP 1	x64Win64VS2008
		Sun Java Platform Standard Edition JDK 1.6	x64Win64jdk

[a] On Windows XP: If you are using JDK 5.0 and want to use Intel's HyperThreading technology, use JDK 5.0 Update 6 (build 1.5.0_06), which includes fixes to JNI and HyperThreading. (If you must use Update 5 (build 1.5.0_05), you should disable HyperThreading.)

[b] The RTI .NET assemblies are supported for both the C++/CLI and C# languages. The type support code generated by rtiddsgen is in C++/CLI; compiling the generated type support code requires Microsoft Visual C++. Calling the assembly from C# requires Microsoft Visual C#.

[c] Windows XP does not support IP_TOS unless registry changes are made. See http://support.microsoft.com/kb/248611, http://www.microsoft.com/technet/technetmag/issues/2007/02/CableGuy/default.aspx.

The compiler flags and the libraries you will need to link into your application are listed in the following tables:

- ❏ Windows XP Professional x64 Edition: Table 10.3 on page 64
- ❏ All other supported Windows platforms: Table 10.2 on page 62. (See also: Libraries Required for Using RTI Secure WAN Transport APIs (Section 10.9).)

To use libraries that are *statically* linked into an application, link in all of the libraries listed in one of the rows of these tables. To use *dynamic* link libraries (DLL) on Windows systems, link in all of the libraries listed in one of the 'Dynamic' sections of the appropriate table. When the application executes, it will attempt to dynamically link in the libraries, which are located in the directory **$(NDDSHOME)\lib\<*architecture*>** (this directory must be placed on the path before the executable is started).

Windows libraries are provided in formats with and without debugging symbols. Choose the format appropriate for your current work. Do not mix libraries built for different formats.

Visual Studio® 2005 — Service Pack 1 Redistributable Package MFC Security Update Requirement

- ❏ You must have the Microsoft Visual C++ 2005 Service Pack 1 Redistributable Package MFC Security Update installed on the machine where you are *running* an application built with the release or debug libraries of the following RTI architecture packages:
  - i86Win32VS2005 and x64Win64VS2005, built with dynamic libraries
  - i86Win32jdk and x64Win64jdk
  - i86Win32dotnet2.0 and x64Win64dotnet2.0

  The Microsoft Visual C++ 2005 Service Pack 1 Redistributable Package MFC Security Update can be obtained from the following Microsoft website:

  - http://www.microsoft.com/download/en/details.aspx?id=26347

Visual Studio® 2008 — Service Pack 1 Requirement

❑ You must have Visual Studio 2008 Service Pack 1 or the Microsoft Visual C++ 2008 SP1 Redistribution Package installed on the machine where you are *running* an application built with the following RTI architecture packages:

- x64Win64VS2008 built with dynamic libraries
- i86Win32VS2008 built with dynamic libraries

The Microsoft Visual C++ 2008 SP1 Redistribution Package can be downloaded from the following Microsoft website:

- For x86 architectures: http://www.microsoft.com/downloads/details.aspx?familyid=A5C84275-3B97-4AB7-A40D-3802B2AF5FC2&displaylang=en
- For x64 architectures: http://www.microsoft.com/downloads/details.aspx?FamilyID=ba9257ca-337f-4b40-8c14-157cfdffee4e&displaylang=en

Visual Studio 2010 — Service Pack 1 Requirement

❑ You must have Visual Studio 2010 Service Pack 1 or the Microsoft Visual C++ 2010 SP1 Redistribution Package installed on the machine where you are *running* an application built with the release libraries of the following RTI architecture packages:

- i86Win32VS2010 built with dynamic libraries
- x64Win64VS2010 built with dynamic libraries
- i86Win32dotnet4.0 and x64Win64dotnet4.0

To run an application built with *debug* libraries of the above RTI architecture packages, you must have Visual Studio 2010 Service Pack 1 installed.

The Microsoft Visual C++ 2010 Service Pack 1 Redistribution Package can be obtained from the following Microsoft website:

- For x86 architectures:

  http://www.microsoft.com/download/en/details.aspx?id=5555
- For x64 architectures:

  http://www.microsoft.com/download/en/details.aspx?id=14632

Windows Registry Setting for Better Performance:

On all Windows systems *prior to* Windows Vista, the following registry setting change will improve performance when sending UDP datagrams of size larger than 1024 bytes:

Under **HKEY_LOCAL_MACHINE, SYSTEM, CurrentControlSet, Services, AFD, Parameters**, add the following:

DWORD: Name=FastSendDatagramThreshold, Value = 65536

This will improve the *Connext* performance for data sizes larger than 1024 bytes (RTPS overhead included). It allows the datagrams to bypass the I/O subsystem by using a blocking send call instead of a buffer copy in the Windows Network stack.

Table 10.4 on page 65 provides details on the environment variables required to be set at run time for a Windows architecture.

For details on how the libraries were built by RTI, see Table 10.5 on page 66. This information is provided strictly for informational purposes; you do not need to use these parameters to compile your application. You may find this information useful if you are involved in any in-depth debugging.

Table 10.6 on page 68 and Table 10.7 on page 69 list additional libraries required when using the optional *RTI Secure WAN Transport* and *RTI TCP Transport*, respectively.

10.1 Use Dynamic MFC Library, Not Static

To avoid communication problems in your *Connext* application, use the dynamic MFC library, not the static version.

If you use the static version, your *Connext* application may stop receiving samples once the Windows sockets are initialized.

10.2 Visual Studio 2005 Required when Using RTI 'Debug' Libraries for Java or .NET APIs

The *Connext* dynamic libraries for Java and .NET[1] rely on Microsoft Visual Studio 2005 Service Pack 1 run-time libraries. These libraries are available with Microsoft Visual Studio 2005 or as part of a redistributable package independent of Visual Studio. (The redistributable package is available for download from the RTI Customer Portal.)

However, while Microsoft includes debug versions of these run-time libraries with Visual Studio, it only includes *release* versions in the redistributable package. This limitation means that if you do not have Visual Studio 2005 installed, you cannot use the RTI *debug* libraries; you must use the RTI *release* libraries. If you attempt to use the RTI debug libraries, and your system does not have debug versions of the Microsoft run-time libraries available, your application will fail to start up properly. If you start it from a command shell, you will see an error about a failure to load the dynamic libraries.

Fortunately, you do not need to use the RTI debug libraries to debug your own code. If you experience library-loading problems when your Java or .NET application starts up in debug mode, modify your application project files to use the release versions of the RTI libraries. Alternatively, you can obtain a no-cost version of Visual Studio 2005 directly from Microsoft, which will contain the necessary debug libraries.

10.3 .NET API Requires Thread Affinity

To maintain proper concurrency control, .NET threads that call a *Connext* API must correspond one-to-one with operating system threads. In most applications, this will always be the case. However, it may not be the case if the threads you are using are managed in a more advanced way—for example, Microsoft SQL Server does this, or you may do so in your own application.

If you intend to call *Connext* APIs from explicitly managed threads, you must first call **Thread.BeginThreadAffinity()** in each such thread to ensure that it remains attached to a single operating system thread. See http://msdn.microsoft.com/en-us/library/system.threading.thread.beginthreadaffinity.aspx.

When you are done making RTI calls from a given thread, you should call **Thread.EndThreadAffinity()**.

In any case, be sure to consult the RTI API documentation for more information about the thread safety contracts of the operations you use.

10.4 Multicast Support

Multicast is supported on all platforms and is configured out of the box. That is, the default value for the initial peers list (**NDDS_DISCOVERY_PEERS**) includes a multicast address. See the online documentation for more information.

[1] *RTI Connext* .NET language binding is currently supported for C# and C++/CLI.

10.5 Supported Transports

Shared memory: Shared memory is supported and enabled by default. The Windows operating system manages the shared memory resources automatically. Cleanup is not required.

UDPv4: Supported and enabled by default.

UDPv6: Supported but disabled on architectures that use Visual Studio 2003 or higher. The peers list (**NDDS_DISCOVERY_PEERS**) must be modified to support UDPv6. No Traffic Class support.

TCP/IPv4: Supported on architectures that use Visual Studio 2005 or higher. (This is *not* a built-in transport.)

10.6 Monotonic Clock Support

The monotonic clock (described in Section 8.6 in the *RTI Core Libraries and Utilities User's Manual*) is supported.

10.7 Support for Controlling CPU Core Affinity for RTI Threads

Support for controlling CPU core affinity (described in Section 19.5 in the *RTI Core Libraries and Utilities User's Manual*) is not available for Windows platforms.

10.8 PPP Link Support for Windows XP Systems

To use a Windows XP point-to-point protocol (PPP) link (such as a serial cable), the UDP transport properties for the *Connext* applications running on the PPP server machine *must* be configured with multicast disabled for the PPP server interface(s).

To disable multicast for an interface, change the UDPv4 transport properties as follows:

```
// Disable multicast for PPP interface because it causes problems:
char *bad_interfaces[] = { "192.168.250.100"}; // interface addr
const int num_bad_interfaces =
    sizeof(bad_interfaces)/sizeof(bad_interfaces[0]);
UDPv4Properties.parent.deny_multicast_interfaces_list =
    bad_interfaces;
UDPv4Properties.parent.deny_multicast_interfaces_list_length =
    num_bad_interfaces;
```

Failure to do so will result in *Connext* being unable to send any data at all over the PPP link.

Notes:

- Setting up multicast-related socket options for the PPP interface can prevent future *unicast* sends using that socket from working.
- *Connext* sets up certain sockets for multicast even if it has no multicast peers, in case some show up later. You avoid this by configuring the multicast deny list as described above.

10.9 Libraries Required for Using RTI Secure WAN Transport APIs

This section is only relevant if you have installed *RTI Secure WAN Transport.* This feature is not part of the standard *Connext* package. If you choose to use it, it must be downloaded and installed separately. It is only available on specific architectures. See the *RTI Secure WAN Transport Release Notes* and *RTI Secure WAN Transport Installation Guide* for details.

To use the WAN or Secure Transport APIs, add the libraries from Table 10.6 on page 68 to your project files.

10.10 Libraries Required for Using RTI TCP Transport APIs

To use the TCP Transport APIs, link against the additional libraries from Table 10.7 on page 69. (Select the files appropriate for your chosen library format.)

Table 10.2 **Building Instructions for Windows Host Architectures**

API	Library Format	RTI Libraries or Jar Files[a]	Required System Libraries	Required Compiler Flags
C	Static Release	nddscz.lib nddscorez.lib For *Connext Messaging*, also include: rticonnextmsgcz.lib	netapi32.lib advapi32.lib user32.lib ws2_32.lib	/D "RTI_WIN32" /MT
	Static Debug	nddsczd.lib nddscorezd.lib For *Connext Messaging*, also include: rticonnextmsgczd.lib		/D "RTI_WIN32" /MTd
	Dynamic Release	nddsc.lib nddscore.lib For *Connext Messaging*, also include: rticonnextmsgc.lib		/D "RTI_WIN32" /D "NDDS_DLL_VARIABLE" /MD
	Dynamic Debug	nddscd.lib nddscored.lib For *Connext Messaging*, also include: rticonnextmsgcd.lib		/D "RTI_WIN32" /D "NDDS_DLL_VARIABLE" /MDd
C++	Static Release	nddscppz.lib nddscz.lib nddscorez.lib For *Connext Messaging*, also include: rticonnextmsgcppz.lib	netapi32.lib advapi32.lib user32.lib ws2_32.lib	/D "RTI_WIN32" /MT
	Static Debug	nddscppzd.lib nddsczd.lib nddscorezd.lib For *Connext Messaging*, also include: rticonnextmsgcppzd.lib		/D "RTI_WIN32" /MTd
	Dynamic Release	nddscpp.lib nddsc.lib nddscore.lib For *Connext Messaging*, also include: rticonnextmsgcpp.lib		/D "RTI_WIN32" /D "NDDS_DLL_VARIABLE" /MD
	Dynamic Debug	nddscppd.lib nddscd.lib nddscored.lib For *Connext Messaging*, also include: rticonnextmsgcppd.lib		/D "RTI_WIN32" /D "NDDS_DLL_VARIABLE" /MDd

Table 10.2 **Building Instructions for Windows Host Architectures**

API	Library Format	RTI Libraries or Jar Files[a]	Required System Libraries	Required Compiler Flags
C++/ CLI	Release	nddscpp.lib nddsc.lib nddscore.lib nddsdotnet.dll or nddsdotnet40.dll	N/A	/D "RTI_WIN32" /D "NDDS_DLL_VARIABLE" /MD /D "WIN32_LEAN_AND_MEAN"
	Debug	nddscppd.lib nddscd.lib nddscored.lib nddsdotnetd.dll or nddsdotnet40.dll		/D "RTI_WIN32" /D "NDDS_DLL_VARIABLE" /MDd /D "WIN32_LEAN_AND_MEAN"
C#	Release	nddsdotnet.dll or nddsdotnet40.dll For *Connext Messaging*, also include: rticonnextdotnet.dll or rticonnextdotnet40.dll	N/A	N/A
	Debug	nddsdotnetd.dll or nddsdotnet40d.dll For *Connext Messaging*, also include: rticonnextdotnetd.dll or rticonnextdotnet40d.dll		
Java	Release	nddsjava.jar For *Connext Messaging*, also include: rticonnextmsg.jar	N/A	N/A
	Debug	nddsjavad.jar For *Connext Messaging*, also include: rticonnextmsgd.jar		

[a] The RTI C/C++ libraries are located in $(NDDSHOME)\lib\<*architecture*>\.
The RTI Java libraries are located in $(NDDSHOME)\class\.
(where $(NDDSHOME) is where *Connext* is installed, such as c:\rti\ndds.5.0.x)

Table 10.3 **Building Instructions for Windows Target Architectures**

API	Library Format	RTI Libraries or Jar Files[a]	Required System Libraries	Required Compiler Flags
C	Static Release	nddscz.lib nddscorez.lib For *Connext Messaging*, also include: rticonnextmsgcz.lib	netapi32.lib advapi32.lib user32.lib ws2_32.lib	/Gd /MT /D "WIN32" /D "RTI_WIN32" /D "NDEBUG"
	Static Debug	nddsczd.lib nddscorezd.lib For *Connext Messaging*, also include: rticonnextmsgczd.lib		/Gd /MTd /D "WIN32" /D "RTI_WIN32"
	Dynamic Release	nddsc.lib nddscore.lib For *Connext Messaging*, also include: rticonnextmsgc.lib		/Gd /MD /D "WIN32" /D "NDDS_DLL_VARIABLE" /D "RTI_WIN32" /D "NDEBUG"
	Dynamic Debug	nddscd.lib nddscored.lib For *Connext Messaging*, also include: rticonnextmsgcd.lib		/Gd /MDd /D "WIN32" /D "NDDS_DLL_VARIABLE" /D "RTI_WIN32"
C++	Static Release	nddscppz.lib nddscz.lib nddscorez.lib For *Connext Messaging*, also include: rticonnextmsgcppz.lib	netapi32.lib advapi32.lib user32.lib ws2_32.lib	/Gd /EHsc /MT /D "WIN32" /D "RTI_WIN32" /D "NDEBUG"
	Static Debug	nddscppzd.lib nddsczd.lib nddscorezd.lib For *Connext Messaging*, also include: rticonnextmsgcppzd.lib		/Gd /EHsc /MTd /D "WIN32" /D "RTI_WIN32"
	Dynamic Release	nddscpp.lib nddsc.lib nddscore.lib For *Connext Messaging*, also include: rticonnextmsgcpp.lib		/Gd /EHsc /MD /D "WIN32" /D "NDDS_DLL_VARIABLE" /D "RTI_WIN32" /D "NDEBUG"
	Dynamic Debug	nddscppd.lib nddscd.lib nddscored.lib For *Connext Messaging*, also include: rticonnextmsgcppd.lib		/Gd /EHsc /MDd /D "WIN32" /D "NDDS_DLL_VARIABLE" /D "RTI_WIN32"

Table 10.3 **Building Instructions for Windows Target Architectures**

API	Library Format	RTI Libraries or Jar Files[a]	Required System Libraries	Required Compiler Flags
C#	Release	nddsdotnet.dll or nddsdotnet40.dll For *Connext Messaging*, also include: rticonnextdotnet.dll or rticonnextdotnet40.dll	N/A	N/A
	Debug	nddsdotnetd.dll or nddsdotnet40d.dll For *Connext Messaging*, also include: rticonnextdotnetd.dll or rticonnextdotnet40d.dll		
C++/ CLI	Release	nddscpp.lib nddsc.lib nddscore.lib	netapi32.lib advapi32.lib user32.lib ws2_32.lib	/Gd /EHsc /MD /D “WIN32” /D “NDDS_DLL_VARIABLE” /D “RTI_WIN32” /D “NDEBUG”
	Debug	nddscppd.lib nddscd.lib nddscored.lib		/Gd /EHsc /MDd /D “WIN32” /D “NDDS_DLL_VARIABLE” /D “RTI_WIN32”
Java	Release	nddsjava.jar For *Connext Messaging*, also include: rticonnextmsg.jar	N/A	N/A
	Debug	nddsjavad.jar For *Connext Messaging*, also include: rticonnextmsgd.jar		

[a] The RTI C/C++ libraries are located in $(NDDSHOME)\lib\<*architecture*>\.
The RTI Java libraries are located in $(NDDSHOME)\class\.
(where $(NDDSHOME) is where *Connext* is installed, such as c:\rti\ndds.5.0.x

Table 10.4 **Running Instructions for Windows Architectures**

RTI Architecture	Library Format	Environment Variables
All supported Windows architectures for Java	N/A	Path=%NDDSHOME%\lib\<*architecture*>; %Path% [a]
All other supported Windows architectures	Static (Release and Debug)	None required
	Dynamic (Release and Debug)	Path=%NDDSHOME%\lib\<*architecture*>; %Path%[a]

[a] %NDDSHOME% represents the root directory of your *Connext* installation. %Path% represents the value of the Path variable prior to changing it to support *Connext*. When using nddsjava.jar, the Java virtual machine (JVM) will attempt to load release versions of the native libraries. When using nddsjavad.jar, the JVM will attempt to load debug versions of the native libraries.

Table 10.5 **Library-Creation Details for Windows Host Architectures**

RTI Architecture	Library Format	Compiler Flags Used by RTI
i86Win32dotnet2.0, i86Win32dotnet4.0	Dynamic Release	/O2 /GL /D "WIN32" /D "NDEBUG" /D "NDDS_DLL_VARIABLE" /D "_WINDLL" /D "_UNICODE" /D "UNICODE" /FD /EHa /MD /c /Zi /clr /TP
	Dynamic Debug	/Od /D "WIN32" /D "_DEBUG" /D "NDDS_DLL_VARIABLE" /D "_WINDLL" /D "_UNICODE" /D "UNICODE" /FD /EHa /MDd /c /Zi /clr /TP
i86Win32jdk	Dynamic Release	-target 1.4 –source 1.4
	Dynamic Debug	-target 1.4 –source 1.4 -g
i86Win32VS2005	Static Release	-DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=I80586 -DTARGET=\"i86Win32VS2005\" -DWIN32 -D_WINDOWS -D_WIN32_WINNT=0x0501 -DWIN32_LEAN_AND_MEAN /O2 /Zi /MT /EHsc -D_CRT_SECURE_NO_DEPRECATE -DNDEBUG -c
	Dynamic Release	-DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=I80586 -DTARGET=\"i86Win32VS2005\" -DWIN32 -D_WINDOWS -D_WIN32_WINNT=0x0501 -DWIN32_LEAN_AND_MEAN /O2 /Zi /MD /EHsc -D_CRT_SECURE_NO_DEPRECATE -DNDEBUG -c
	Static Debug	-DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=I80586 -DTARGET=\"i86Win32VS2005\" -DWIN32 -D_WINDOWS -D_WIN32_WINNT=0x0501 -DWIN32_LEAN_AND_MEAN /Od /ZI /MTd /EHsc /RTC1 -D_CRT_SECURE_NO_DEPRECATE -c
	Dynamic Debug	-DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=I80586 -DTARGE T=\"i86Win32VS2005\" -DWIN32 -D_WINDOWS -D_WIN32_WINNT=0x0501 -DWIN32_LEAN_AND_MEAN /Od /ZI /MDd /EHsc /RTC1 -D_CRT_SECURE_NO_DEPRECATE -c
i86Win32VS2008	Static Release	-DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=I80586 -DTARGET=\"i86Win32VS2008\" -DWIN32 -D_WINDOWS -D_WIN32_WINNT=0x0501 -DWIN32_LEAN_AND_MEAN /O2 /Zi /MT /EHsc -D_CRT_SECURE_NO_DEPRECATE -DNDEBUG -c
	Dynamic Release	-DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=I80586 -DTARGET=\"i86Win32VS2008\" -DWIN32 -D_WINDOWS -D_WIN32_WINNT=0x0501 -DWIN32_LEAN_AND_MEAN /O2 /Zi /MD /EHsc -D_CRT_SECURE_NO_DEPRECATE -DNDEBUG -c
	Static Debug	-DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=I80586 -DTARGET=\"i86Win32VS2008\" -DWIN32 -D_WINDOWS -D_WIN32_WINNT=0x0501 -DWIN32_LEAN_AND_MEAN /Od /ZI /MTd /EHsc /RTC1 -D_CRT_SECURE_NO_DEPRECATE -c
	Dynamic Debug	-DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=I80586 -DTARGE T=\"i86Win32VS2008\" -DWIN32 -D_WINDOWS -D_WIN32_WINNT=0x0501 -DWIN32_LEAN_AND_MEAN /Od /ZI /MDd /EHsc /RTC1 -D_CRT_SECURE_NO_DEPRECATE -c

Table 10.5 **Library-Creation Details for Windows Host Architectures**

RTI Architecture	Library Format	Compiler Flags Used by RTI
i86Win32VS2010	Static Release	-DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=I80586 -DTARGET=\"i86Win32VS2010\" -DWIN32 -D_WINDOWS -D_WIN32_WINNT=0x0501 -DWIN32_LEAN_AND_MEAN /O2 /Zi /MT /EHsc -D_CRT_SECURE_NO_DEPRECATE -DNDEBUG -c
	Dynamic Release	-DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=I80586 -DTARGET=\"i86Win32VS2010\" -DWIN32 -D_WINDOWS -D_WIN32_WINNT=0x0501 -DWIN32_LEAN_AND_MEAN /O2 /Zi /MD /EHsc -D_CRT_SECURE_NO_DEPRECATE -DNDEBUG -c
	Static Debug	-DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=I80586 -DTARGET=\"i86Win32VS2010\" -DWIN32 -D_WINDOWS -D_WIN32_WINNT=0x0501 -DWIN32_LEAN_AND_MEAN /Od /ZI /MTd /EHsc /RTC1 -D_CRT_SECURE_NO_DEPRECATE -c
	Dynamic Debug	-DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=I80586 -DTARGE T=\"i86Win32VS2010\" -DWIN32 -D_WINDOWS -D_WIN32_WINNT=0x0501 -DWIN32_LEAN_AND_MEAN /Od /ZI /MDd /EHsc /RTC1 -D_CRT_SECURE_NO_DEPRECATE -c
x64Win64dotnet2.0, x64Win64dotnet4.0	Dynamic Release	/O2 /GL /D "WIN64" /D "NDEBUG" /D "NDDS_DLL_VARIABLE" /D "_WINDLL" /D "_UNICODE" /D "UNICODE" /FD /EHa /MD /c /Zi /clr /TP
	Dynamic Debug	/Od /D "WIN64" /D "_DEBUG" /D "NDDS_DLL_VARIABLE" /D "_WINDLL" /D "_UNICODE" /D "UNICODE" /FD /EHa /MDd /c /Zi /clr /TP
x64Win64jdk	Dynamic Release	-target 1.4 –source 1.6
	Dynamic Debug	-target 1.4 –source 1.6 -g
x64Win64VS2005 Note: linker requires /MACHINE:X64 option.	Static Release	/W3 -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=AMD64 -DTARGET=\"x64Win64VS2005\" -DWIN32 -D_WINDOWS -D_WIN32_WINNT=0x0501 -DWIN32_LEAN_AND_MEAN /O2 /Zi /MT /EHsc -D_CRT_SECURE_NO_DEPRECATE -DNDEBUG -c
	Dynamic Release	/W3 -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=AMD64 -DTARGET=\"x64Win64VS2005\" -DWIN32 -D_WINDOWS -D_WIN32_WINNT=0x0501 -DWIN32_LEAN_AND_MEAN /O2 /Zi /MD /EHsc -D_CRT_SECURE_NO_DEPRECATE -DNDEBUG -c
	Static Debug	/W3 -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=AMD64 -DTARGET=\"x64Win64VS2005\" -DWIN32 -D_WINDOWS -D_WIN32_WINNT=0x0501 -DWIN32_LEAN_AND_MEAN /Od /ZI /MTd /EHsc /RTC1 -D_CRT_SECURE_NO_DEPRECATE -c
	Dynamic Debug	/W3 -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=AMD64 -DTARGET=\"x64Win64VS2005\" -DWIN32 -D_WINDOWS -D_WIN32_WINNT=0x0501 -DWIN32_LEAN_AND_MEAN /Od /ZI /MDd /EHsc /RTC1 -D_CRT_SECURE_NO_DEPRECATE -c

Table 10.5 **Library-Creation Details for Windows Host Architectures**

RTI Architecture	Library Format	Compiler Flags Used by RTI
x64Win64VS2008 Note: linker requires / MACHINE:X64 option.	Static Release	/W3 -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=AMD64 -DTARGET=\"x64Win64VS2008\" -DWIN32 -D_WINDOWS -D_WIN32_WINNT=0x0501 -DWIN32_LEAN_AND_MEAN /O2 /Zi /MT /EHsc -D_CRT_SECURE_NO_DEPRECATE -DNDEBUG -c
	Dynamic Release	/W3 -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=AMD64 -DTARGET=\"x64Win64VS2008\" -DWIN32 -D_WINDOWS -D_WIN32_WINNT=0x0501 -DWIN32_LEAN_AND_MEAN /O2 /Zi /MD /EHsc -D_CRT_SECURE_NO_DEPRECATE -DNDEBUG -c
	Static Debug	/W3 -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=AMD64 -DTARGET=\"x64Win64VS2008\" -DWIN32 -D_WINDOWS -D_WIN32_WINNT=0x0501 -DWIN32_LEAN_AND_MEAN /Od /ZI /MTd /EHsc /RTC1 -D_CRT_SECURE_NO_DEPRECATE -c
	Dynamic Debug	/W3 -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=AMD64 -DTARGET=\"x64Win64VS2008\" -DWIN32 -D_WINDOWS -D_WIN32_WINNT=0x0501 -DWIN32_LEAN_AND_MEAN /Od /ZI /MDd /EHsc /RTC1 -D_CRT_SECURE_NO_DEPRECATE -c
x64Win64VS2010 Note: linker requires / MACHINE:X64 option.	Static Release	/W3 -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=AMD64 -DTARGET=\"x64Win64VS2010\" -DWIN32 -D_WINDOWS -D_WIN32_WINNT=0x0501 -DWIN32_LEAN_AND_MEAN /O2 /Zi /MT /EHsc -D_CRT_SECURE_NO_DEPRECATE -DNDEBUG -c
	Dynamic Release	/W3 -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=AMD64 -DTARGET=\"x64Win64VS2010\" -DWIN32 -D_WINDOWS -D_WIN32_WINNT=0x0501 -DWIN32_LEAN_AND_MEAN /O2 /Zi /MD /EHsc -D_CRT_SECURE_NO_DEPRECATE -DNDEBUG -c
	Static Debug	/W3 -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=AMD64 -DTARGET=\"x64Win64VS2010\" -DWIN32 -D_WINDOWS -D_WIN32_WINNT=0x0501 -DWIN32_LEAN_AND_MEAN /Od /ZI /MTd /EHsc /RTC1 -D_CRT_SECURE_NO_DEPRECATE -c
	Dynamic Debug	/W3 -DPtrIntType=long -DCSREAL_IS_FLOAT -DCPU=AMD64 -DTARGET=\"x64Win64VS2010\" -DWIN32 -D_WINDOWS -D_WIN32_WINNT=0x0501 -DWIN32_LEAN_AND_MEAN /Od /ZI /MDd /EHsc /RTC1 -D_CRT_SECURE_NO_DEPRECATE -c

Table 10.6 **Additional Libraries for Using RTI Secure WAN Transport APIs on Windows Systems**

Library Format	RTI Secure WAN Transport Libraries[a]	OpenSSL Libraries[b]
Dynamic Release	nddstransportwan.lib nddstransporttls.lib	ssleay32.lib libeay32.lib
Dynamic Debug	nddstransporttlsd.lib nddstransportwand.lib	
Static Release	nddstransportwanz.lib nddstransporttlsz.lib	
Static Debug	nddstransportwanzd.lib nddstransporttlszd.lib	

[a] These libraries are located in <*wan install dir*>\lib\<*architecture*>\ (where <*wan install dir*> is where *RTI Secure WAN Transport* is installed, such as c:\rti\ndds.5.0.x)

[b] These libraries are located in <*openssl install dir*>\<*architecture*>\lib, where <*openssl install dir*> is where you installed OpenSSL, such as c:\rti\openssl-0.9.8f.

Table 10.7 **Additional Libraries for Using RTI TCP Transport APIs on Windows Systems**

Library Format	RTI TCP Transport Libraries[a]
Dynamic Release	nddstransporttcp.dll
Dynamic Debug	nddstransporttcpd.dll
Static Release	nddstransporttcpz.lib
Static Debug	nddstransporttcpzd.lib

[a] The libraries are located in <*Connext install dir*>\lib\<*architecture*>\, where <*Connext install dir*> is where you installed *Connext*, such as /local/rti/ndds.5.0.x.

Table 10.8 **Additional Libraries for using RTI TCP Transport APIs on Windows Systems with TLS Enabled**

Library Format	RTI TLS Libraries[a]
Dynamic Release	nddstls.dll
Dynamic Debug	nddstlsd.dll
Static Release	nddstlsz.dll
Static Debug	nddstlszd.dll
OpenSSL Libraries	ssleay32.lib libeay32.lib

[a] The libraries are located in <*TLS install dir*>/lib/<*architecture*>/., where <*TLS install dir*> is where you installed *RTI TLS Support*, such as /local/rti/ndds.5.0.x.

11 Custom Supported Platforms

Table 11.1 lists additional target libraries available with *Connext* 5.0, for which RTI offers custom support. If you are interested in using one of these platforms, please contact your local RTI representative or email **sales@rti.com**.

Table 11.1 **Custom Supported Platforms**

Operating System		CPU	Compiler	RTI Architecture Abbreviation
INTEGRITY	INTEGRITY 5.0.11	PPC8349	GHnet2 TCP/IP stack	ppc8349Inty5.0.11.mds8349
Linux	Mistral Linux Kernel 2.6.32	ARMv7	Sourcery G++ Lite 2009q3-67 gcc 4.4.1	armv7leLinux2.6gcc4.4.1
	Red Hat Enterprise Linux 5.1	x86	gcc3.4.6	i86Linux2.6gcc3.4.6
			Sun Java Platform Standard Edition JDK 1.5 and 1.6	i86Linux2.6gcc3.4.6jdk
	Red Hat Enterprise Linux 5.2 (2.6 kernel)	Pentium class	gcc 4.2.1	i86Linux2.6gcc4.2.1
			Sun Java Platform Standard Edition JDK 1.6	i86Linux2.6gcc4.2.1jdk
	Red Hat Enterprise Linux 6 for IBM POWER7 Servers (2.6.32-70.el.ppc64)	POWER7	gcc 4.4.4	power7Linux2.6gcc4.4.4
	RedHawk Linux 5.1	x86	gcc 4.1.2	i86RedHawk5.1gcc4.1.2
			Sun Java Platform Standard Edition JDK 1.5 and 1.6	i86RedHawk5.1gcc4.1.2jdk
	RedHawk Linux 5.4 (2.6 kernel)	Pentium class	gcc 4.2.1	i86RedHawk5.4gcc4.2.1
			Sun Java Platform Standard Edition JDK 1.6	i86RedHawk5.4gcc4.2.1jdk
	RedHawk Linux 6.0	x64	gcc 4.4.5	x64Linux2.6gcc4.4.5
	Wind River Linux 3 (2.6 kernel)	Pentium class	gcc 4.3.2	i86WRLinux2.6gcc4.3.2
OpenVMS	OpenVMS 8.3	IA64	cc 7.3	ia64OpenVMS8_3cc7_3