RTI Connext DDS
Core Libraries
Getting Started Guide
Addendum for INtime Systems
Version 6.1.1
Trademarks

RTI, Real-Time Innovations, Connext, NDDS, the RTI logo, 1RTI and the phrase, “Your Systems. Working as one,” are registered trademarks, trademarks or service marks of Real-Time Innovations, Inc. All other trademarks belong to 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.

This is an independent publication and is neither affiliated with, nor authorized, sponsored, or approved by, Microsoft Corporation.

The security features of this product include software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/). This product includes cryptographic software written by Eric Young (eay@cryptsoft.com). This product includes software written by Tim Hudson (tjh@cryptsoft.com).

Notice

Any deprecations noted in this document serve as notice under the Real-Time Innovations, Inc. Maintenance Policy #4220.

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

Chapter 1 INtime Systems ................................................................. 1

Chapter 2 Installing and Configuring INtime SDK

2.1 Prerequisites ............................................................................. 2

2.2 Installing Connext DDS for INtime Development ............................. 3

Chapter 3 INtime Development Model

3.1 Motivation .................................................................................. 4

3.2 Inter-Node Communication ............................................................. 4

3.3 INtime Software Development Kit .................................................. 4

Chapter 4 Using Connext DDS Utilities on INtime for Windows

4.1 RTI DDS Ping App ....................................................................... 6

4.2 RTI DDS Spy App ......................................................................... 7

4.3 RTI DDS Prototyper App ................................................................. 9

Chapter 5 Using the Connext DDS C and C++ Libraries with INtime on VS 2017

5.1 Setting Up .................................................................................... 10

5.1.1 Environment Variables ............................................................... 10

5.1.2 INtime Kernel Configuration ...................................................... 11

5.2 Building and Running a "Hello, World" Example ............................... 13

5.2.1 Start the Publisher ................................................................. 14

5.2.2 Start the Subscriber ............................................................... 15
Chapter 1 INtime Systems

This document provides instructions to install and configure RTI® Connext® DDS for INtime® for Windows development and run Connext DDS applications on INtime systems. Please refer to the documentation provided by TenAsys® for more information on the INtime operating system: http://www.tenasys.com/support/documentation/.

This section will guide you through the process of generating, compiling, and running a Hello World application on an INtime system by expanding on Hands-On 1 of Introduction to Publish/Subscribe, in the RTI Connext DDS Getting Started Guide.

Chapter 2 Installing and Configuring INtime SDK on page 2 describes how to install and configure Connext DDS for INtime development.

Chapter 3 INtime Development Model on page 4 gives an overview of the INtime development model.

If you want to use the Connext DDS Ping, Spy, or Prototyper utilities, please read Chapter 4 Using Connext DDS Utilities on INtime for Windows on page 6

Chapter 5 Using the Connext DDS C and C++ Libraries with INtime on VS 2017 on page 10 will guide you through the steps to generate, modify, build, and run an example HelloWorld application on an INtime target. You will use rtiddsgen to generate example source code.
Chapter 2 Installing and Configuring INtime SDK

2.1 Prerequisites

1. The development machine needs to meet the following minimum requirements:
   - Pentium class (or better) processor
   - 40MB of free RAM for INtime and your real-time application
   - 250MB hard disk space for tools, examples, and documentation
   - Windows must be loaded using the Windows bootloader. Third-party bootloaders will not load INtime successfully.


3. Install the INtime 6 SDK (Software Development Kit) with the INtime for Windows configuration. For instructions, consult the INtime 6 Quick Start Guide (pp. 5-10), available from http://www.tenasys.com/support/documentation/.

2.2 Installing Connext DDS for INtime Development

You will need to install a host archive and a target archive. The host archive contains documentation, header files, and other files you need for development; the target archive contains the libraries and other files you need for deployment. Unpack them as described in the instructions for Windows systems in the RTI Connext DDS Installation Guide.
Chapter 3 INtime Development Model

INtime is a real-time operating system (RTOS) developed by TenAsys. It can be run in one of two usage configurations:

- **INtime for Windows**: INtime RTOS runs alongside Microsoft Windows OS on the same hardware platform.
- **INtime Distributed RTOS**: INtime runs as a stand-alone RTOS.

This document will only cover development of Connext DDS applications for the INtime for Windows configuration.

### 3.1 Motivation

**INtime for Windows** is intended for use in applications that incorporate both feature-rich and real-time components. Feature-rich components like machine-user interfaces, graphical representation, mathematical modeling, and I/O support benefit from running in a Windows environment, while real-time components like advanced CNC, PLC, medical equipment, and robotics run on the INtime RTOS. **INtime for Windows** manages communication between these applications, allowing them to interact at the process level.

### 3.2 Inter-Node Communication

As described above, **INtime for Windows** allows INtime and Windows to run simultaneously on the same hardware platform. **INtime for Windows** partitions system resources between the two environments to ensure that they do not conflict with each other and remain independent. An INtime partition is known as a "node" in the INtime environment.

### 3.3 INtime Software Development Kit

All development is performed with the INtime SDK running on a Windows PC and comes as a complete solution with tools and utilities to facilitate the development, deployment, and field
Chapter 4 Using Connext DDS Utilities on INtime for Windows

You can use the Connext DDS rtiddsping, rtiddsspy, and rtiddsprototyper utilities on INtime for Windows through the File Explorer or Command Prompt.

4.1 RTI DDS Ping App

To use rtiddsping on INtime:

1. Load and run the INtime 6 Kernel on your system. From Windows desktop, click on the INtime icon on the right-hand side of the taskbar and select "Start <node>".
2. Open the file rtiddsping.rta. This can be done with File Explorer or Command Prompt.

   **With File Explorer:** navigate to $NDDSHOME/bin and double click on rtiddsping.rta.

   **With Command Prompt:** change directories to $NDDSHOME/bin, as shown in the example below:

   ```
   C:\> cd $NDDSHOME/bin
   ```

   Then call rtiddsping.rta:

   ```
   C:\Program Files\rti_connext_dds-6.1.1\bin> rtiddsping
   ```

   **NOTE:** If a window appears with the following error message, make sure that the INtime 6 Kernel is running on your system; see step 1.
3. INtime will start running rtiddsping and open an INtime Console window.

Data from rtiddsping is displayed in the INtime Console window.

### 4.2 RTI DDS Spy App

To use rtiddspy on INtime:

1. Load and run the INtime 6 Kernel on your system. From Windows desktop, click on the INtime icon on the right-hand side of the taskbar and select "Start <node>".
2. Open the file rtiddspy.rta. This can be done with File Explorer or Command Prompt.

   **With File Explorer:** navigate to $NDDSHOME/bin and double click on rtiddspy.rta.

   **With Command Prompt:** change directories to $NDDSHOME/bin, as shown in the example below:

   ```
   C:\> cd $NDDSHOME/bin
   ```

   Then call rtiddspy.rta:
4.2 RTI DDS Spy App

C:\Program Files\rti_connext.dds-6.1.1\bin> rtiddsspy

**NOTE:** If a window appears with the following error message, make sure that the INtime 6 Kernel is running on your system; see step 1.

3. INtime will start running *rtiddsspy* and open an INtime Console window.

Data from *rtiddsspy* is displayed in the INtime Console window. If *rtiddsping* is running simultaneously, *rtiddsspy* should receive and display matching data. For more information on running *rtiddsping*, see 4.1 RTI DDS Ping App on page 6.
4.3 RTI DDS Prototyper App

To use rtiddsprototyper on INtime:

1. Load and run the INtime 6 Kernel on your system. From Windows desktop, click on the INtime icon on the right-hand side of the taskbar and select "Start <node>".

2. Open the file `rtiddsprototyper.rta`. This can be done with File Explorer or Command Prompt.

   **With File Explorer:** navigate to `SNDDSHOME/bin` and double click on `rtiddsprototyper.rta`.

   **With Command Prompt:** change directories to `SNDDSHOME/bin`, as shown in the example below:

   ```
   C:\> cd $NDDSHOME/bin
   ```

   Then call `rtiddsprototyper.rta`:

   ```
   C:\Program Files\rti_connext.dds-6.1.1\bin> rtiddsprototyper
   ```

   **NOTE:** If a window appears with the following error message, make sure that the INtime 6 Kernel is running on your system; see step 1.

3. INtime will start running `rtiddsprototyper`. 
Chapter 5 Using the Connext DDS C and C++ Libraries with INtime on VS 2017

This section describes how to build a Connext DDS application with C++ for an INtime target. The process for building a Connext DDS application with C is similar to that for C++. The following instructions and screenshots are for C++; any differences in the instructions for C will be called out in the affected steps. Also note that when developing with C, instructions that include the text “Cpp” should be replaced with “C”, such as in filenames, App names, etc. For example, HelloCppSubscriber for C++ would be HelloCSubscriber for C.

5.1 Setting Up

There are a few steps to take before building and running a Hello World application for INtime.

5.1.1 Environment Variables

The NDDSHOME environment variable should be set as described in Set Up Environment Variables (rtisetenv), in "Hands-On 1" of Introduction to Publish/Subscribe, in the RTI Connext DDS Getting Started Guide.

1. Create a new environment variable called INTIME.

2. Set the INTIME environment variable to your INtime installation directory. By default, the INtime root installation path is C:\Program Files (x86)\INtime\ as shown in the figure below.
5.1.2 INtime Kernel Configuration

The INtime Kernel, by default, will search for dynamic libraries in pre-defined paths. The location of the RTI dynamic libraries must be added to the list of paths that INtime will search.

1. Click on the INtime Kernel node in the task bar.

2. Select INtime Configuration.

4. Select the applicable nodes from the left panel.
5. Select the System tab.
6. In the System tab, select the field "Search RSL Path". Click on the right-hand side of the row.

7. Click New. Add the path `<NDDSHOME>/lib\i86INtime6.3VS2017` to the RSL search list, substituting `<NDDSHOME>` with your installation path to Connext DDS. Click OK.
5.2 Building and Running a "Hello, World" Example

This section describes the basic steps for building and running an `rtiddsgen`-generated "Hello, World" example for an INtime for Windows system target.

1. Create a directory to work in. In this example, we use a directory called `myhello`.
2. In the `myhello` directory, create a file called `HelloWorld.idl` that contains a user-defined data type:

   ```
   struct HelloWorld {
     string<128> msg;
   }
   ```

3. Use the `rtiddsgen` utility to generate sample code.

   Open Command Prompt, change directories to the location of `HelloWorld.idl`, and enter the following:

   ```
   rtiddsgen -language C++ -example i86INtime6.3VS2017 HelloWorld.idl
   ```

4. `rtiddsgen` will output a Microsoft Visual Studio solution file with the extension `.sln`. Open this solution in Visual Studio; it will automatically open in the INtime perspective within Visual Studio.

---

**NOTE:** The **NDDSHOME** variable **cannot** be used in the path you specify in the RSL search list; the full system path **must** be given. If **Connext DDS** is installed at the default location, the system path will match the one shown in the figure below.
5.2.1 Start the Publisher

You should see that there are two projects: **HelloWorld_publisher** and **HelloWorld_subscriber**.

5. Select the required build mode from the drop-down menu in the toolbar. There are 4 available modes: **Release**, **Debug**, **Release RSL**, and **Debug RSL**.

6. Select **Build** from the menu bar.

7. Click on **Build Solution**.

**5.2.1 Start the Publisher**

From your command shell, go to examples\connext.dds\c++\hello_world and type:

```
> piperta objs\i86INtime6.3VS2017\HelloWorld_publisher.rta
```

Note that **i86INtime6.3VS2017** is the architecture name for **INtime for Windows**; see the contents of the examples directory for a list of available architectures.
5.2.2 Start the Subscriber

From your command shell, go to `examples\connext.dds\c++\hello_world` and type:

```bash
> piperta objs\i86INtime6.3VS2017\HelloWorld_subscriber.rta
```

Note that `i86INtime6.3VS2017` is the architecture name for *InTime for Windows*; see the contents of the `examples` directory for a list of available architectures.

You should be able to see communication between `HelloWorld_publisher` and `HelloWorld_subscriber` while both are running.

**Note:** If you are using a USER_QOS_FILE to set QoS properties, be sure to place this file in the same directory as the .rta executable so that the QoS file is picked up correctly at runtime.