



Product:	DiamondSystemsIO
Product version:	v1.1
Document ID:	PM-DiamondSystemsLib
Doc revision:	A
Written/Aprr.:	RE / SL
Date:	3. Oct. 2008

## Industrial Control Design AS



# DiamondSystemsIO v1.1

## Programmers Manual

The content of this document is confidential information not to be published without the consent of Industrial Control Design AS.

Industrial Control Design AS, [www.icd.no](http://www.icd.no), [support@icd.no](mailto:support@icd.no), [forum.icd.no](http://forum.icd.no)

# Contents

---

<b>1. INTRODUCTION.....</b>	<b>3</b>
1.1. About DiamondSystemsIO.....	3
<hr/>	
<b>2. INSTALLATION.....</b>	<b>4</b>
2.1. Prerequisites.....	4
2.2. Quick setup of a project utilizing the DiamondSystemsIO library.....	4
2.3. Install on a Controller.....	8

# 1. Introduction

## 1.1. About DiamondSystemsIO

The DiamondSystemsIO component provides access to the Diamond Systems DMM 16 AT analog / digital card through the DiamondSystemsIO CDP Component. The DiamondSystemsIO is an IO Server based CDP Component that retrieves and sets digital and analog signal values from and to the card. If several cards are connected to the same CDP controller, one DiamondSystemsIO server must exist for each card. If you have two cards, you must have two DiamondSystemsIO servers.

***Warning: On system power-up (time from booting until CDP is completely up and running), the digital outputs will be set to a low (value 0), while the Analog outputs might fluctuate. Due to this, essential control outputs should be controlled (enabled) by a high signal from the digital outputs to prevent problems when powering up/down the controller.***

***An Autocalibrate feature, which assumes that the digital outputs are set to a low value to disable the controlled equipment might also be present, so for this reason, beware of how the board is connected.***

This document describes how to set up and use the DiamondSystemIO with the CDP system. The DiamondSystemIO component has the following features:

- Supports the DMM-16-AT card
- Works under On-Time RTOS32.
- Converts CDP Signals to physical signals
- Converts Physical signals into CDP Signals
- Has Online/Offline State for checking communication-problems.

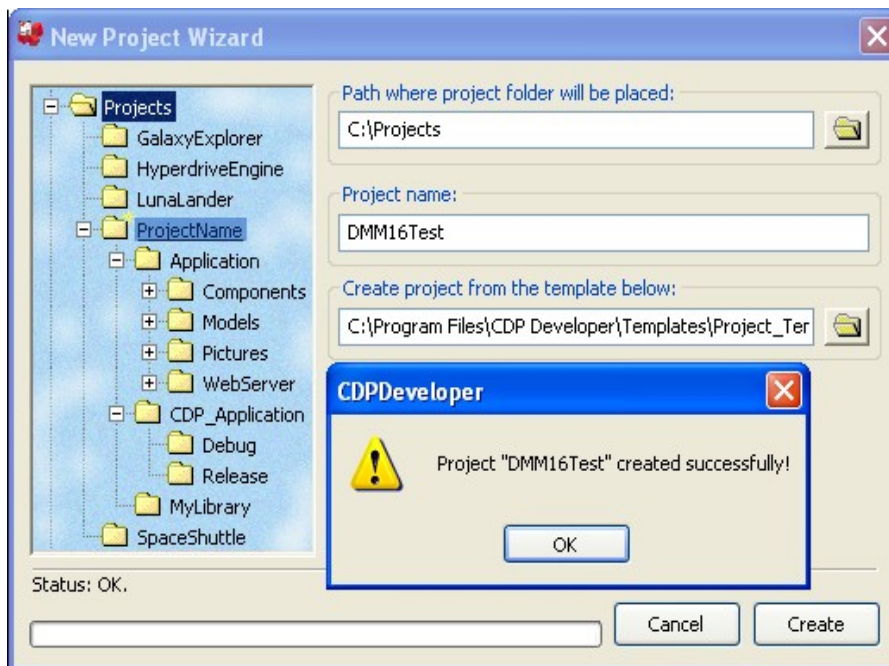
# 2. Installation

## 2.1. Prerequisites

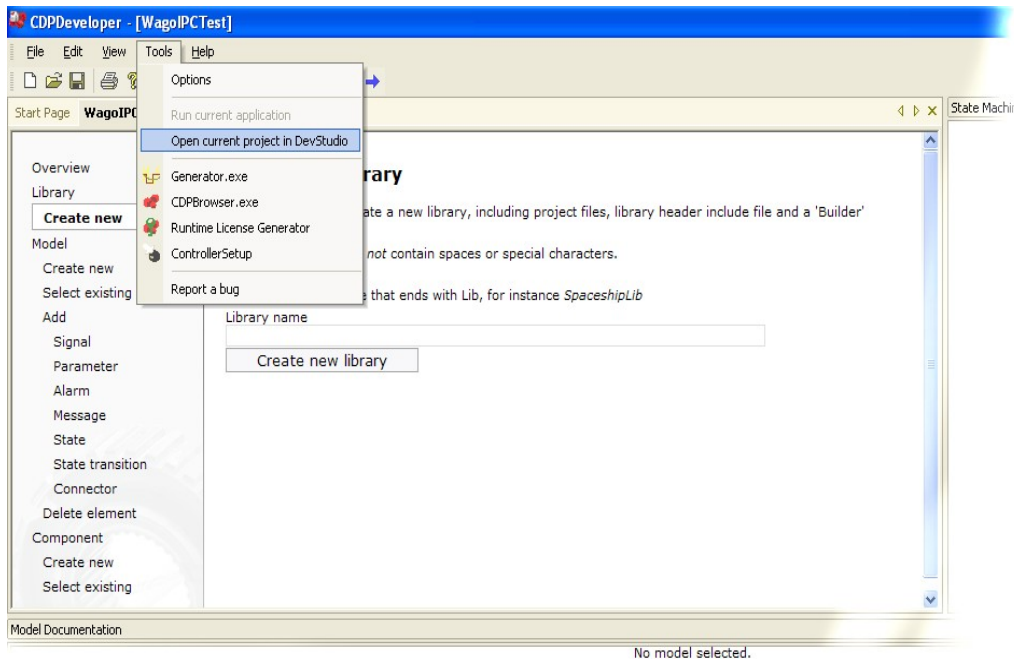
- A valid CDP license
- You are familiar with CDP
- CDP version 2.3.1.0 for OnTime RTOS32, and OnTime RTOS32 version 5.09 is installed
- XMLParser add-on is installed
- DiamondSystems DMM 16 AT card

## 2.2. Quick setup of a project utilizing the DiamondSystemsIO library

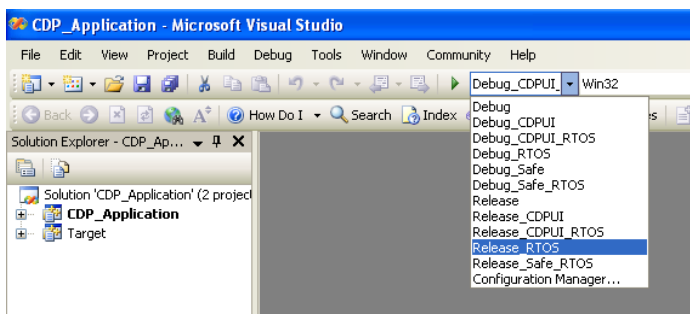
1. Start CDPDeveloper
2. Make a new project by Selecting File->New, type in project name and click Create (or skip to step 4 and open an already existing project in Visual Studio)



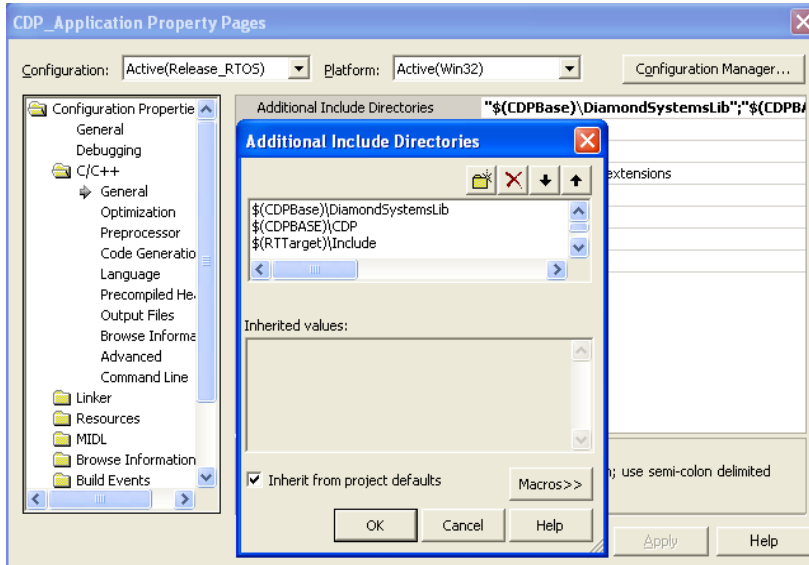
3. Choose 'Open current project in Devstudio' from the Tools menu:



4. If Visual Studio asks to convert the project, accept this by selecting 'Next'/'Finish'/'Close' until done. Close the conversion report.
5. In Visual Studio, select the 'Release\_RTOS' configuration.

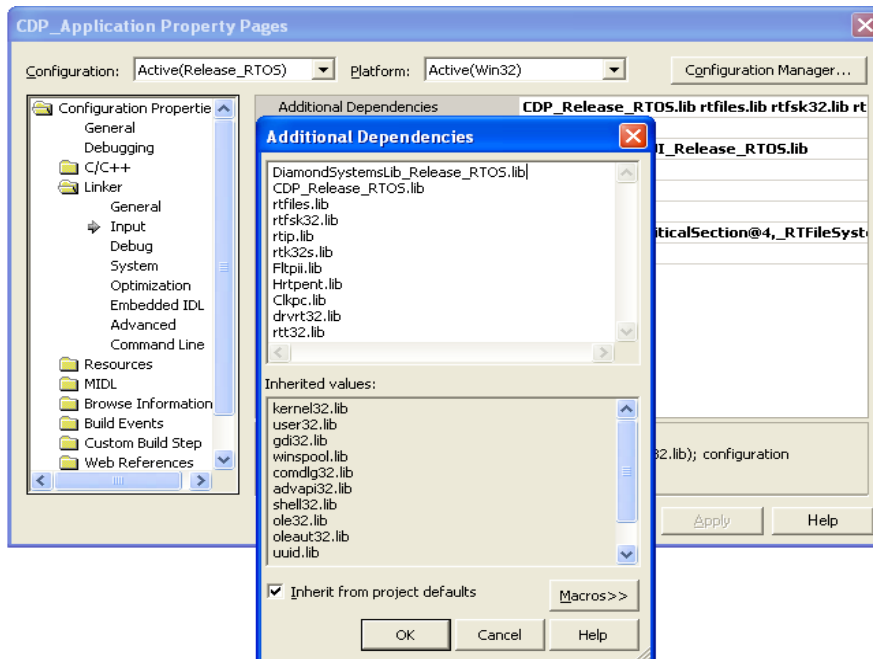


6. Select 'CDP\_Application' from the 'Solution Explorer', right-click and select Properties.
7. In C++/Additional include Directories, make sure it says:  
`“$(CDPBase)\DiamondSystemsLib”;“$(CDPBase)\CDP”;“$(RTTarget)\Include”`

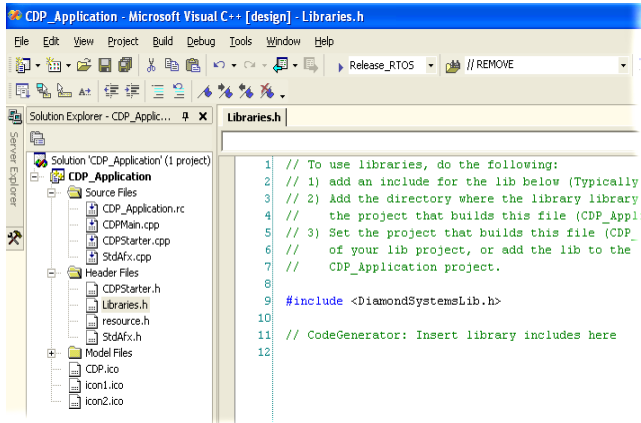


The compiler will look in the directories specified in 'Additional Include Directories' for files that you #include in your .cpp and .h files. If you get an error 'Can not open include file ...' then it is most likely caused by a missing include directory, or that the file you #include does not exist.

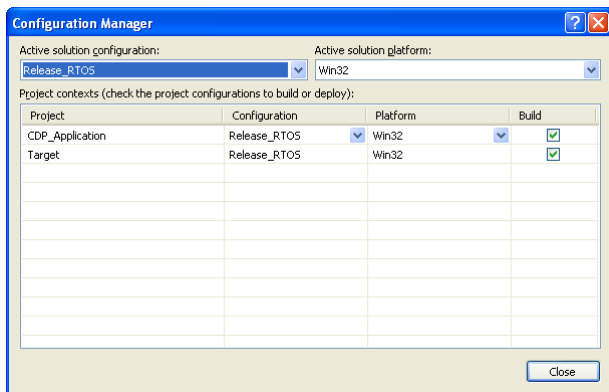
8. In Linker->Input, make sure you list DiamondSystemsLib\_Release\_RTOS.lib CDP\_Release\_RTOS.lib in addition to all the on-time rtos32 libraries you need. This will ensure that the linker finds all the functions that is referenced in the code.



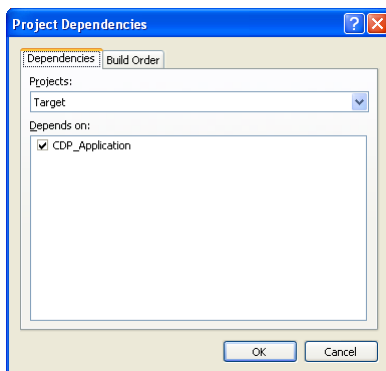
9. In the 'Solution Explorer', inside the 'CDP\_Application' project, locate the Header file 'Libraries.h', and add the line
10. #include <DiamondSystemsLib.h>:



11. If there is no 'Target' project in your 'Solution Explorer' Select 'Solution' -> 'Add' -> 'Existing Project'. Choose the 'Target.vcproj' from the same folder in which your 'CDP\_Application.vcproj' is at.
12. Right-Click the Target project, select Properties. Under General, make sure 'Configuration Type' says 'Utility'. Click OK.
13. Right-Click Solution, select 'Configuration manager...'. Make sure that all configurations have 'Release\_RTOS' selected in 'Configuration' :



14. Choose 'Solution' -> 'Project Dependencies', choose 'Target' from the pull-down list, check 'CDP\_Application' and click OK.



15. Rebuild the entire solution to generate the CDP.rtb file in the CDP\_Application\Release\_RTOS\ folder.

## 2.3. Install on a Controller

Assuming you have a Controller with PXE boot and a disk, you can use ControllerSetup.exe to download a bare-bone CDP application to the controller. See the document 'ControllerSetup.pdf' in the Doc folder where you installed CDP for more information on setting up a controller.

When the Controller has a CDP Application installed, you can use CDPFileManager.exe to upload files to the controller. Upload these files to the root of the controller disk.

- CDP\_Application\Release\_RTOS\CDP.rtb
- Everything inside the Application folder.

Make sure that you also upload rttboot.com from your “\$(RTTarget)\bin” folder to the root folder of the controller. This ensures that the RTOS32 loader is the most recent version.