System 800xA Control

AC 800M
Getting Started

System Version 6.0

Power and productivity
for a better world™

ABB

System 800xA Control

AC 800M
Getting Started

System Version 6.0

NOTICE

This document contains information about one or more ABB products and may include a description of or a reference to one or more standards that may be generally relevant to the ABB products. The presence of any such description of a standard or reference to a standard is not a representation that all of the ABB products referenced in this document support all of the features of the described or referenced standard. In order to determine the specific features supported by a particular ABB product, the reader should consult the product specifications for the particular ABB product.

ABB may have one or more patents or pending patent applications protecting the intellectual property in the ABB products described in this document.

The information in this document is subject to change without notice and should not be construed as a commitment by ABB. ABB assumes no responsibility for any errors that may appear in this document.

Products described or referenced in this document are designed to be connected, and to communicate information and data via a secure network. It is the sole responsibility of the system/product owner to provide and continuously ensure a secure connection between the product and the system network and/or any other networks that may be connected.

The system/product owners must establish and maintain appropriate measures, including, but not limited to, the installation of firewalls, application of authentication measures, encryption of data, installation of antivirus programs, and so on, to protect the system, its products and networks, against security breaches, unauthorized access, interference, intrusion, leakage, and/or theft of data or information.

ABB verifies the function of released products and updates. However system/product owners are ultimately responsible to ensure that any system update (including but not limited to code changes, configuration file changes, third-party software updates or patches, hardware change out, and so on) is compatible with the security measures implemented. The system/product owners must verify that the system and associated products function as expected in the environment they are deployed.

In no event shall ABB be liable for direct, indirect, special, incidental or consequential damages of any nature or kind arising from the use of this document, nor shall ABB be liable for incidental or consequential damages arising from use of any software or hardware described in this document.

This document and parts thereof must not be reproduced or copied without written permission from ABB, and the contents thereof must not be imparted to a third party nor used for any unauthorized purpose.

The software or hardware described in this document is furnished under a license and may be used, copied, or disclosed only in accordance with the terms of such license. This product meets the requirements specified in EMC Directive 2004/108/EC and in Low Voltage Directive 2006/95/EC.

TRADEMARKS

All rights to copyrights, registered trademarks, and trademarks reside with their respective owners.

Release: April 2016
Document number: 3BSE041880-600 A

TABLE OF CONTENTS

Section 3 - MyDoors Project

Section 4 - Hardware Configuration

Section 5 - Connecting the Controller and Go Online

Appendix B - License Management

Appendix C - SIL Certified Applications

Appendix D - Communication Cables

INDEX

About This User Manual

⚠ Any security measures described in this User Manual, for example, for user access, password security, network security, firewalls, virus protection, etc., represent possible steps that a user of an 800xA System may want to consider based on a risk assessment for a particular application and installation. This risk assessment, as well as the proper implementation, configuration, installation, operation, administration, and maintenance of all relevant security related equipment, software, and procedures, are the responsibility of the user of the 800xA System.

Welcome to Control Builder Professional for AC 800M. This manual is produced for anyone intending to use the programming tool Control Builder for the first time. It is focused on getting you quickly started and acquainted with the product. Therefore, as much 'in-depth' information as possible has been separated from the main sections and placed in appendices instead.

If this is your first time working with the programming tool, it is recommended that you start by reading Section 1, Introduction and then work yourself through each section.

The sections are organized in this manner:

Section 1, Introduction, gives you an overview of the product Control Builder.

Section 2, Control Builder User Interface is a brief introduction to the Control Builder's core interface Project Explorer.

Section 3, MyDoors Project, encourages you to build a small project example to get yourself acquainted with the Control Builder environment.

Section 4, Hardware Configuration, teaches you how to add hardware units to your project.

Section 5, Connecting the Controller and Go Online, starts with the prerequisites for connecting a controller (correct firmware and system identity) and then guide you through downloading a project and Go online.

Section 6, View Live Data in Plant Explorer will introduce you to the Plant Explorer interface and help you studying how variable values from MyDoors project are transferred from a controller, via an OPC Server, up as live data in a Workplace.

User Manual Conventions

Microsoft Windows conventions are normally used for the standard presentation of material when entering text, key sequences, prompts, messages, menu items, screen elements, etc.

Feature Pack

The Feature Pack content (including text, tables, and figures) included in this User Manual is distinguished from the existing content using the following two separators:

Feature Pack Functionality_____

<Feature Pack Content>

_____

Feature Pack functionality included in an existing table is indicated using a table footnote (*) :
*Feature Pack Functionality

Feature Pack functionality in an existing figure is indicated using callouts.

Unless noted, all other information in this User Manual applies to 800xA Systems with or without a Feature Pack installed.

Warning, Caution, Information, and Tip Icons

This User Manual includes Warning, Caution, and Information where appropriate to point out safety related or other important information. It also includes Tip to point out useful hints to the reader. The corresponding symbols should be interpreted as follows:

Electrical warning icon indicates the presence of a hazard that could result in *electrical shock.*

Warning icon indicates the presence of a hazard that could result in *personal injury.*

Caution icon indicates important information or warning related to the concept discussed in the text. It might indicate the presence of a hazard that could result in *corruption of software or damage to equipment/property.*

Information icon alerts the reader to pertinent facts and conditions.

Tip icon indicates advice on, for example, how to design your project or how to use a certain function

Although Warning hazards are related to personal injury, and Caution hazards are associated with equipment or property damage, it should be understood that operation of damaged equipment could, under certain operational conditions, result in degraded process performance leading to personal injury or death. Therefore, fully comply with all Warning and Caution notices.

Terminology

A complete and comprehensive list of terms is included in *System 800xA System Guide Functional Description (3BSE038018*)*. The listing includes terms and definitions that apply to the 800xA System where the usage is different from commonly accepted industry standard definitions and definitions given in standard dictionaries such as Webster's Dictionary of Computer Terms.

Section 1 Introduction

General

Control Builder Professional (CB Professional) is a programming tool for creating control solutions when using the AC 800M hardware.

The Control Builder comes with type solutions for simple logic control, device control, loop control, alarm handling, and so on, packaged as standard libraries.

It provides a wizard function for hardware definition files, support for multi-user engineering, and an evaluation mode for testing and evaluating new applications against a running application.

Control Builder supports five different programming languages according to IEC 61131-3. They are Function Block Diagram, Structured Text, Instruction List, Ladder Diagram and Sequential Function Chart.
In addition to these, Control Builder supports creation of logic using Diagrams (which use the Function Diagram (FD) language) and Control Module Diagrams. Other useful functionality are configuring High Integrity controllers for SIL applications, online debugger, test and simulation mode, and so on.

Product Overview

Control Builder is a fully integrated control system function in the 800xA system. It provides tools for programming applications and configure hardware units from the AC 800M family.

It is accessed through the Project Explorer interface, and runs on any of the following platforms:

- US English version of Windows Server 2012 R2 Standard or Datacenter edition

- US English version of Windows 8.1 Professional or Enterprise edition:

    - 64-bit (x64)

800xA System

Project Explorer	AC 800M Aspect system	Plant Explorer
Handle one control project		**Handle many control projects**
Build control project		**Administrate control projects**
Download to AC 800M controller		**Faceplates and alarm lists**
Control system		**Operator workplace**
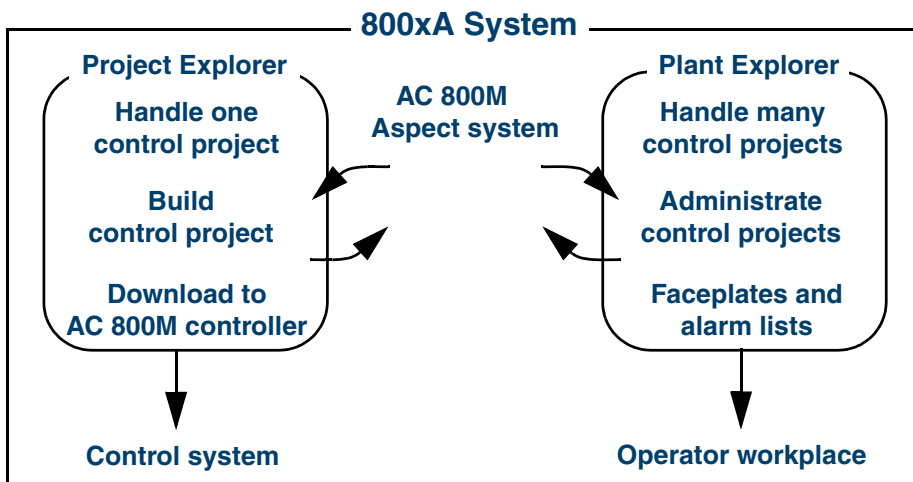
Figure 1. Project Explorer and Plant Explorer are two separate interfaces in Process Portal for building and maintaining control projects.

This section gives an introduction to the Project Explorer interface. Once you are familiar with the basics within this user manual, you are advised to look up the manual *AC 800M Configuration (3BSE035980*)* which describes more thoroughly the 800xA programming functions that can be accessed through the Project Explorer.

> An introduction to the Plant Explorer can be found in the manual, Operator Workplace, Configuration (3BSE030322*).

Project Explorer

The Control Builder user interface is called Project Explorer and this is where you create and build your projects. A project contains the entire configuration needed for a AC 800M based control solution, including control applications and hardware

settings. Context menus are helpful while configuring hardware units or connecting parameters etc. You right-click an object to open its corresponding context menu.

Both the software (programs, diagrams, functions, and so on) and the hardware (the actual hardware connected to the controller) are modelled in a project. The relationships are visualized in Figure 2.

Figure 2. Project Explorer and actual hardware setup.

Libraries

Control Builder is delivered with an extensive set of predefined type solutions stored in standard libraries. These include data types, functions, function block types, control module types, and diagram types, which can be used in your projects.

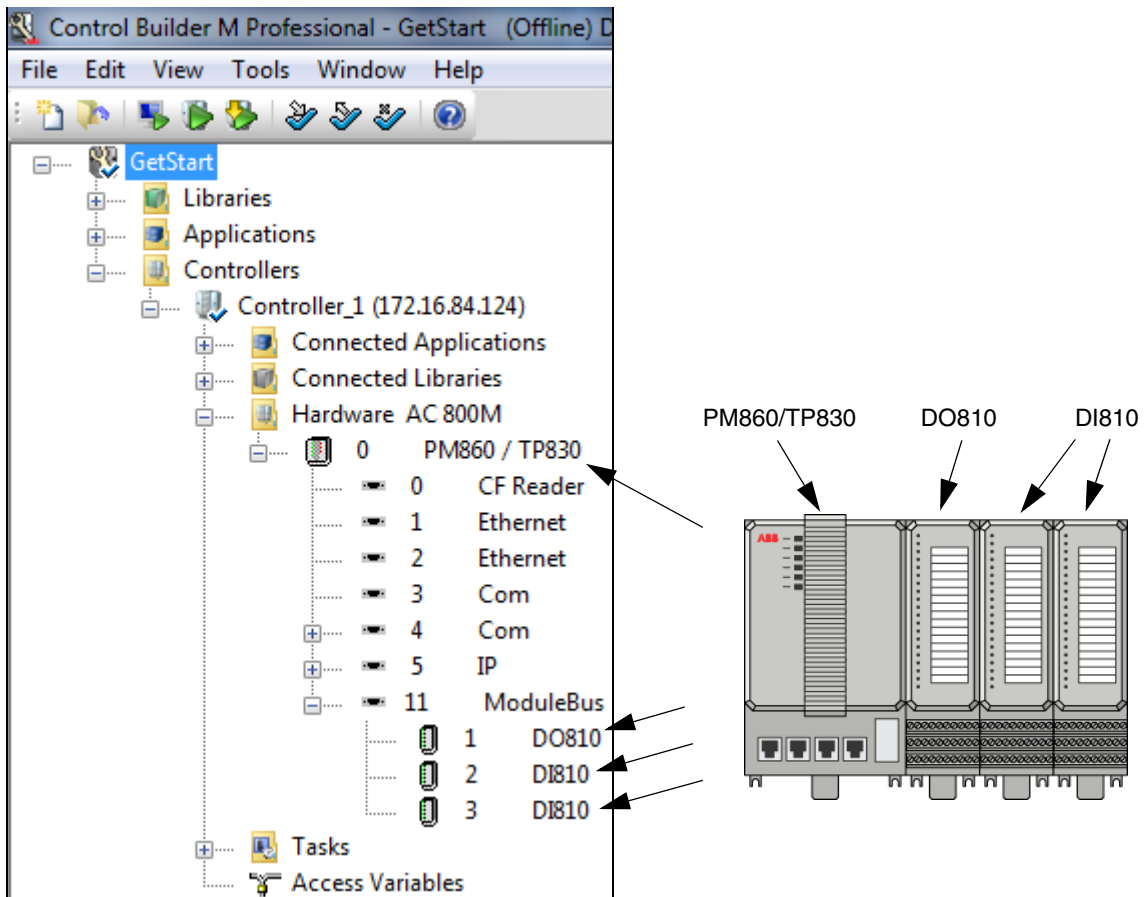All standard libraries are included during the 800xA installation and are available in your projects. The installation provides the following libraries:

- The Basic library, it contains basic building blocks for AC 800M control software like data types, function block types and control module types with extended functionality, designed by ABB. The contents inside the Basic library can be categorized as follows: IEC 61131-3 Function Block Types, Other Function Block Types and Control Module Types.

- The Communication Libraries, they include function blocks for MMS, ModBus RTU, Modbus TCP, Foundation Fieldbus HSE, SattBus, COMLI, UDP and TCP, Siemens 3964R protocols, and IOCommLib for PROFINET.

- The Control Libraries, they include single PID control and cascade PID control function blocks, control modules, etc.

- The Binary Process Libraries, contain types for controlling motors, valves, ABB Drives and Insum Devices. Most types in these libraries are templates, that is, you can copy and modify them to fit your particular process.

- The Alarm and Event Library contain function blocks for alarm and event detection, and alarm printouts on a local printer.

- The Signal Libraries contain types for adding supervision, alarm handling and error handling to I/O signals, and also for the overview and forcing of boolean and real signals.

- The Burner Library contain function block types, which are used for defining logic for Burner control.

A complete list of libraries delivered with the 800xA installation can be found in, *AC 800M Binary and Analog Handling (3BSE035981*)* manual.

Hardware

An extensive set of predefined hardware types, stored in standard hardware libraries, are delivered with Control Builder. These hardware types are used in your projects when configuring the controller hardware.

All standard libraries are included during the 800xA installation and are available in your projects. The installation provides the following libraries:

- The Basic Hardware, it contains basic hardware types for controller hardware, such as types for AC 800M, CPUs, Ethernet communication link, Com port, ModuleBus etc.

- The PROFIBUS Hardwares, they contain hardware types for PROFIBUS communication interfaces, ABB Drives and ABB Panel 800.

- The Communication Hardwares, they contain hardware types for the communication interfaces Foundation Fieldbus HSE, MasterBus 300, Modbus TCP, IEC 61850, PROFIBUS, PROFINET IO, MOD5, AF 100, EtherNet/IP and DeviceNet, INSUM, DriveBus, RS-232C, and UDP and TCP.

- Serial Communication Protocol Hardwares, they contain hardware types for SerialProtocol, COMLI, ModBus and Siemens 3964R.

- The I/O System Hardwares, they include hardware types for I/O communication interfaces, I/O adapters and I/O units; S100 (incl. S100 Rack), S200, S800 and S900.

Control Builder Functions

The Control Builder is used to create control solutions. The solutions are created within control builder projects, and several levels of structuring are available inside one project.

A project in Control Builder can handle up to 1024 applications where each application can handle 64 programs and 128 diagrams at the most. A maximum of 32 Control Builder PCs can be used together in multi-user environment and up to 32 AC 800M controllers can be created and handled within a project.

You can create self-defined libraries containing data types, function block types etc. which can be used in any project.

Besides function block types, your Control Builder can also handle control modules and diagrams, which are components for object-oriented (and graphical) programming.

For more information see also Appendix A, Functions and Settings.

Using Online Help in Control Builder

Control Builder user information comes in three forms:

•   Printed manuals,

•   Online manuals (pdf files, with the same contents as the printed manuals),

•   Control Builder online help.

In addition, there is stand-alone online help for a number of tools.

It is also possible to add help files for non-standard hardware and self-defined libraries. This feature is described in the manual *AC 800M Binary and Analog Handling (3BSE035981*).*

Manuals

All manuals exist in two versions: as a printed manual and as a pdf file. The pdf files (online manuals) are stored on your local disk after installation. Online manuals can be accessed from **ABB Start Menu > ABB Industrial IT 800xA > .....**

This manual, *Getting Started* introduces you to the basic functions necessary to create and download a small control project to a controller. The installation provide a number of additional online manuals in pdf format (many of these can also be ordered as printed manuals):

•   The manual *AC 800M Configuration* describes the basic functions in Control Builder.

•   The manual *AC 800M Planning* describes design issues and programming languages.

- The *Safety Manual*, *AC 800M High Integrity* describes processes and rules that apply when creating SIL certified applications for AC 800M High Integrity controllers.

- The manual *AC 800M, Binary and Analog Handling*, describes the Control Builder standard libraries and how to use them to build complex automation solutions.

- The *AC 800M Controller Hardware* manual describes how to install and configure AC 800M and AC 800M High Integrity controllers, together with AC 800M control software.

- There is also pdf versions of fieldbus and I/O manuals, that is, documentation regarding FOUNDATION Fieldbus, PROFIBUS, PROFINET IO, AF 100, EtherNet/IP and DeviceNet, TRIO, Satt IO, S200 I/O, S800 I/O, and S900 I/O.

Control Builder Online Help

Control Builder Online Help can be accessed in the following ways:

- Context-Sensitive Help (select an object in Project Explorer and press F1; if you press F1 in Plant Explorer, you will open general Plant Explorer help)

- Via the table of contents (select **Help > Help Topics**, then select the Contents tab and click on a topic in the contents tree).

- Via the index (select **Help > Help Topics**, then select the Index tab and enter an index word).

Use the Online Help Index

The index offers a number of ways to find the information you are looking for:

- Enter the action you want information on, for example "configure" or "download".

- Enter the name of the object you want information about, for example "PM864" or "project explorer".

Note that it is not always possible to find information about a single object by entering its name, for example "CI860" or "Level6CC". Try searching for the category instead, for example "I/O units" or "data types". This will normally take you to a list of objects or units, from which you can jump to the one you are interested in.

- Enter the subject you want information on, for example "function block types" or "communication interfaces".

If you are looking for information about a specific library object, or information about a specific hardware unit, the easiest way to find this information is to select the object in Project Explorer and press F1. Control Builder will then take you to the right topic.

Text Search

The text search goes through all topics and shows all the matches, based on the text you enter. For better results, enter the specific text that is relevant to the concept that you are searching.

Before You Begin

This manual assumes that your 800xA system has been installed and configured. The installation procedure is described in the *System 800xA Getting Started (2PAA111708*)* manual.

Three software licenses – Control Builder, SoftController, and Library License are required. For more information on licenses, see Appendix B, License Management.

You need to install OPC Server for AC 800M in order to subscribe to live controller data in Plant Explorer.

Configuration

Before you start, you should have the answer to the following questions:

- Should your engineering workplace consist of a single workplace (single user) or a number of workplaces (multiple users)?

- Should your engineering workplace contain single/multiple environments?

The System Installer will help you:

- Create a system and give it a name.

- Choose whether to run aspect and connectivity servers on the same PC, or separately.

- Define the number of environments.

- Prepare your system for RNRP configuration (if desired).

- Add System Extensions.

- Start the aspect server(s).

- Start the system.

Refer to the System Installer described in the *System 800xA Getting Started (2PAA111708*)* manual.

Configure OPC Server

For information on how to configure an OPC Server, refer to the *AC 800M OPC Server (3BSE035983*)* manual. You can read how to connect the OPC Server in subsection OPC Server on page 89.

Section 2 Control Builder User Interface

Introduction

The core interface of Control Builder is the Project Explorer. After getting familiarized with Project Explorer in this section, you can refer to Section 3, MyDoors Project, to create a new project and its logic.

About Project and its Objects

In a Control Builder project, the available objects for creating control logic are applications, programs, diagrams, control modules, and function blocks. The following list describes the hierarchy of these and their usage, in descending order:

• An automation system might contain a number of control networks representing different parts of a large plant.

• Within each control network, you can create a number of *projects*.

• A project is the top level software unit and it contains the configuration data for libraries, applications, connected hardware, etc. It also groups libraries, applications and the connected hardware in an hierarchical tree structure in Project Explorer.

• Each application contains *programs, diagrams* or *control modules*, and additional objects (data types, function block types, control module types, diagram types) that are used within the application.

• Each program, diagram or control module is connected to a task, which decides how often each of them is executed. It is also possible to connect individual function blocks and control modules, which are used to build the objects, to different tasks.

A complete control network is only represented in Plant Explorer, and not in Project Explorer. Therefore the control network is not illustrated here in Figure 3. The sequence below shows the steps from creating a new project to a download. As you can see, a Project is the highest level object in Control Builder, whereas a program, diagram or control module is an alternative to handle code below inside an application.
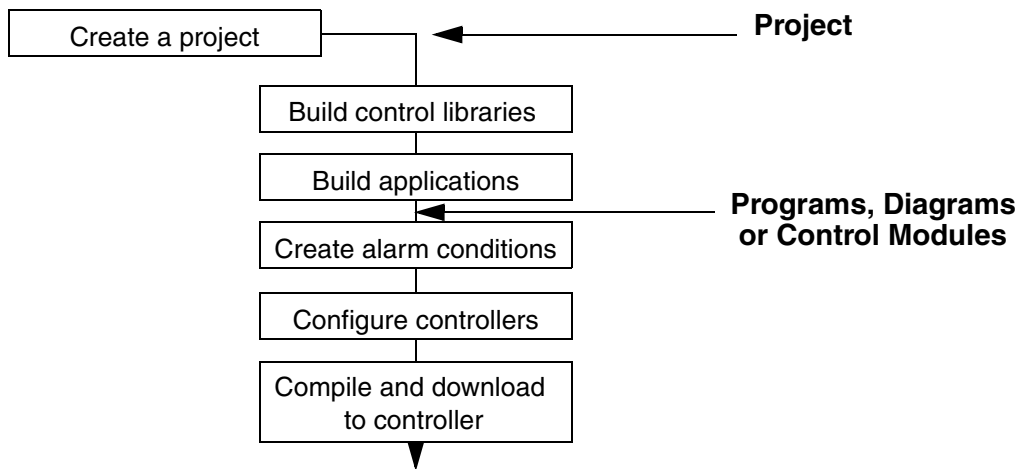

Figure 3. Sequence for building a project and the hierarchy between a project and programs, diagrams or control modules.

About Entities and Reservation

Several users can work at the same time with a Control Builder project. To avoid that different users edit the same object properties at the same time, you have to reserve the relevant object, for example a project or an application, before you can modify its properties. This ensures that only one user can modify an object at a time.

When ready, you release the reservation. Now, it is possible for another user to reserve and modify the object.

However, only some objects, so-called entities, can be reserved. An entity is a set of objects and aspects that is reserved as a single unit.

• Examples of objects that are entities: projects, libraries, applications, diagrams, diagram types, control module types, function block types and controllers.

- Examples of object that are no entities: programs and datatypes.

To edit an object that is no entity (for example a program), you first have to reserve the parent entity (in this example, an application).

For more information on entities and reservation of entities, refer to the manual *AC 800M Configuration (3BSE035980*).*

About Environments

Environments require a separate license and are not available to all users. The Project Explorer will only show information on environments when they are used.

Environments provide isolated engineering. Since different environments can have different contents, you can modify a control application without affecting the running application.

When you create a new project, or open an existing one, you must first select the relevant environment. The Control Builder project will then be opened in that environment. When you modify an object in one environment, the changes will only be visible in that environment. However, users working in the same environment will see each others changes.

You can also transfer the changes from one environment to another. This process is called **deploy**.

The two basic environments are:

- Engineering Environment is used for engineering tasks, for example to modify a project, or an application.

- Production Environment is used to compile and download a project (or a single application) to the controller and go online.

For more information, refer to the *System 800xA, Configuration (3BDS011222*)* and to the *AC 800M Configuration (3BSE035980*)* manuals.

Project Templates

When a new project is to be created, the Control Builder provides a set of predefined templates. These templates contain predefined initial setup data, suitable for different kind of projects.

The following project templates are available in the Control Builder:

- AC800M
    - For normal use, and for running non-SIL applications)

- AC800M_HighIntegrity_SM811
    - For running non-SIL, SIL1-2, and SIL3 applications

- AC800M_HighIntegrity_SM812
    - For running non-SIL, SIL1-2, and SIL3 applications.

- EmptyProject[1]
    - Rarely used, and has a minimum configuration with initial data only under the Libraries folder.

- SoftController
    - For development use, and for simulating non-SIL applications without a controller.

- SoftController_HI
    - For development use, and for simulating SIL applications without a controller,

SIL stands for "Safety Integrity Level". For more information see Appendix C, SIL Certified Applications.

---

1. An empty project template contains only the compulsory system firmware functions, with no additional application or hardware functions.

Project Explorer

Project Explorer is the main interface to the Control Builder programming tool. It displays the currently active control project.

> **i** Only one project can be opened at a time in the Project Explorer, although all the projects can be viewed in the Plant Explorer.
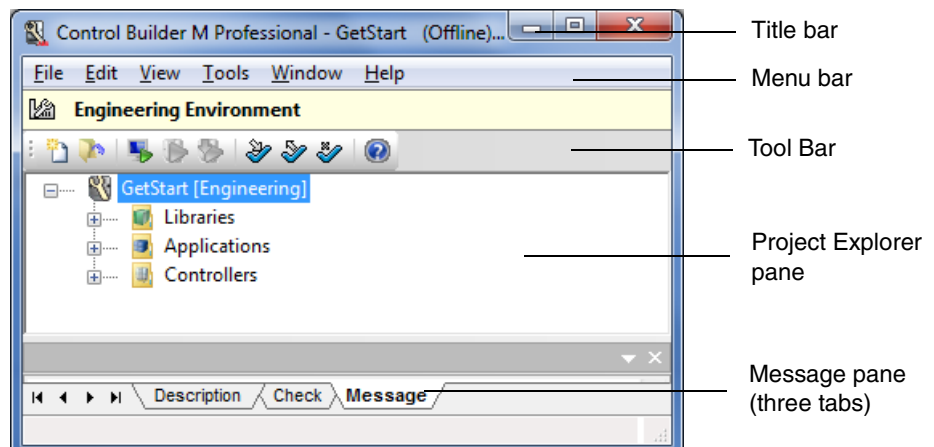

Figure 4. Project Explorer.

Title Bar, Menu Bar and Tool Bar

The title bar of the Control Builder shows the name of the current project, the status of the project (Offline, TestMode, or Online), and the name of the 800xA system.

> **i** When using environments, the Information bar shows relevant environment. For example, the project in Figure 4 is opened in the Engineering Environment.

The menu bar contains the drop-down menus: File, Edit, View, Tools, Window, and Help. When the menu items on the menus are dimmed, they cannot be accessed (the function is not allowed in the current context).

The tool bar contains icons that serve as shortcuts to the most common Control Builder functions, such as Download, Reserve, and Online help.

Project Explorer Pane

The Project Explorer pane contains three main folders, see Figure 5:

- The Libraries folder, see Libraries Folder on page 29.

- The Applications folder, see Applications Folder on page 31.

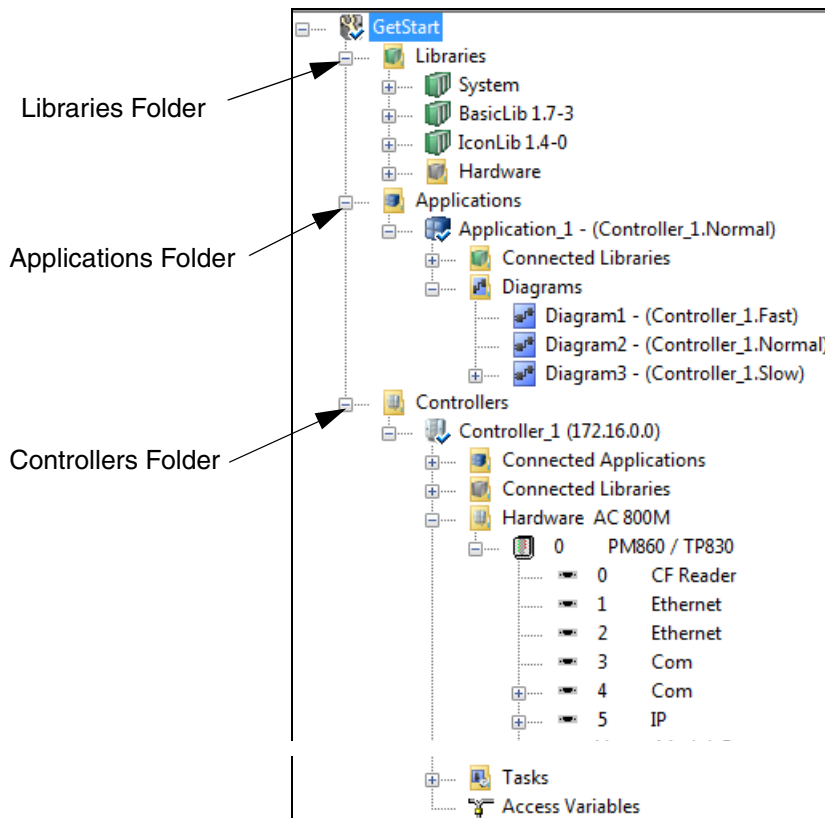- The Controllers folder, see Controllers Folder on page 34.

Figure 5. The Project Explorer pane, showing the three main folders Libraries, Applications, and Controllers.

Libraries Folder

When a project is created, the Libraries folder contains the System folder (containing firmware functions that can be used throughout the applications), and two libraries that are always connected to a project: the BasicLib and the IconLib.

Besides these three libraries, the Libraries folder also contains the Hardware folder with the library containing the basic hardware types:

- BasicHWLib (if AC800M, EmptyProject, or SoftController template is used for creating the project).

- BasicHIHwLib (if AC800M_HighIntegrity or SoftController_HI template is used for creating the project).

After the project is created, both standard libraries and user-defined libraries can be created or inserted into the Libraries folder and the Hardware folder.

When a new library is created, the subfolders – Data Types, Diagram Types, Control Module Types, and Function Block Types – are not visible since they do not contain any objects.

Right-click the library to open the context menu, and go to **New** which displays the option to create the different types (see Figure 6). Once the types are created, they are displayed under the corresponding subfolders in the library.
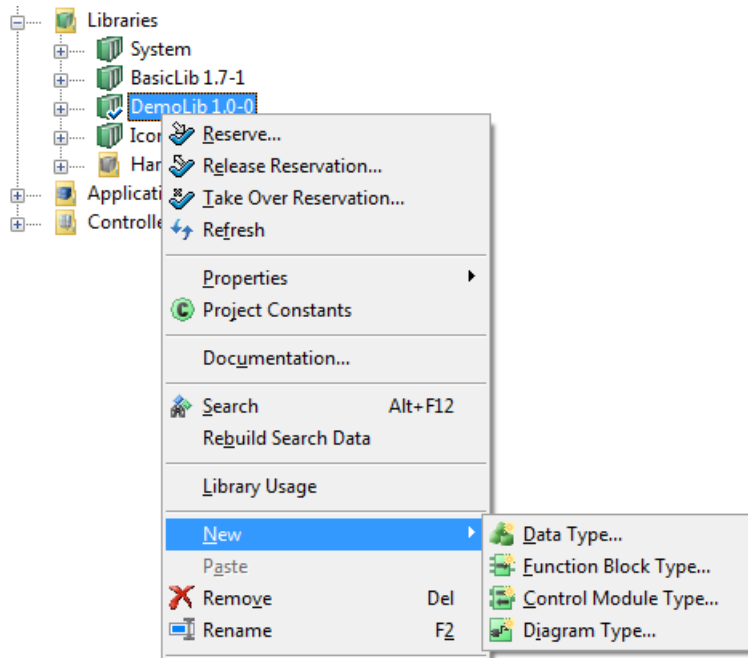
Figure 6. Creating types under a library

A library can only be added to an application if it has first been added to the
Libraries folder. A hardware unit (type) can only be connected in a controller
configuration if the corresponding hardware library is added to the Hardware folder.

Libraries can only be added to the Libraries and Hardware folders if they exist in the
Library Structure in Plant Explorer.

> For more information on libraries and library handling, see the manual *AC 800M,*
> *Configuration* and the manual *AC 800M, Binary and Analog Handling*.

Applications Folder

The Applications folder holds all code that is downloaded to the controller(s). This code is stored as programs, control modules, single control modules or diagrams. The chosen method depends on the requirements of the particular application.

The Applications folder contains applications and other application folders.

To create a new application folder under the Applications folder, right click the Applications folder and select **New Folder** (see Figure 7). The new application folder can in turn contain both applications and application folders.
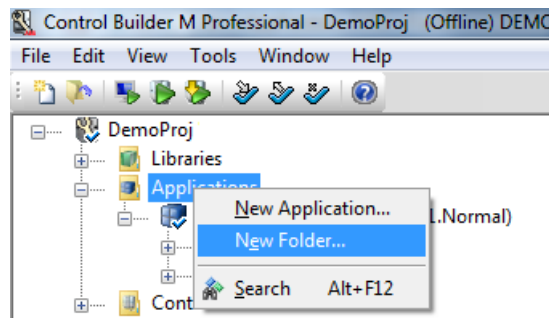

Figure 7. Creating a new application folder inside the Applications folder

The application folders helps to structure or group the applications in the Project Explorer. It is also possible to move applications and application folders in the folder structure using the drag-and-drop operation.

The Connected Libraries folder under the application contains all libraries that are connected to the particular application. To connect a library to an application, right click the Connected Libraries folder, select **Connect Library**, and select the required library from the drop down list.

The types that can be created under the application are Data Types, Diagram Types, Function Block Types, Control Module Types. If a library is connected to the application, the types from that library can also be used in the application.

In the application, the code is organized in any of the following folders:

*   Programs
*   Control Modules
*   Diagrams

For more information, refer to the manual *AC 800M, Configuration.*

The Diagrams folder in the default application contains three diagrams. These three diagrams are connected to three default tasks, see Controllers Folder on page 34. You can change these task connections, as well as add your own tasks and diagrams.

ℹ️ When a new application is created, the subfolders – Programs, Diagrams, Control Modules, Control Module Types, Data Types, Diagram Types, and Function Block Types – are not visible because they do not contain any objects. Right-click the application to open the context menu, and go to **New**, which displays the option to create the different types, control modules, diagrams, and programs (see Figure 8). Once these are created, they are displayed under the corresponding subfolders in the application.
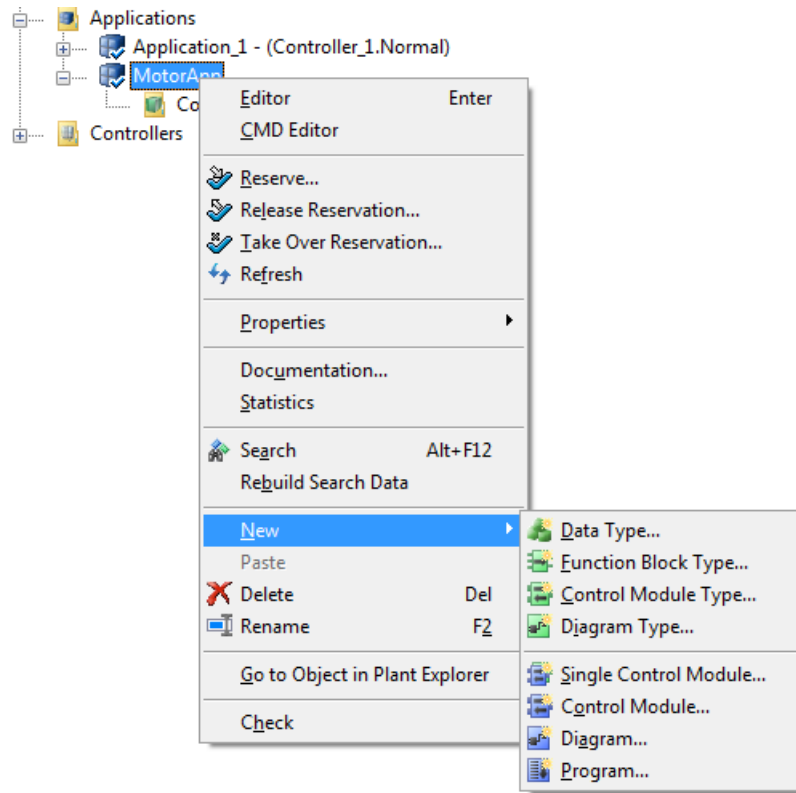
Figure 8. Creating types, control modules, diagrams or programs under an application.

From the objects in the Applications folder, a number of software editors can be opened, see Editors on page 37.

To check the code for errors, click the Check icon on the toolbar. If there are errors in the project, these are indicated by a red triangle next to the object (in Offline mode). The descriptions of the errors are displayed in the Check tab of the message pane.

Controllers Folder

The Controllers folder contains all the controllers that belong to the project.

Each controller has a Connected Applications folder, containing the application(s) running in the controller. The controller also has a Connected Libraries folder, containing all the hardware libraries to be used when configuring the controller.

To connect an application to the controller, right-click the Connected Applications folder, select **Connect Application**, and select the application.

To connect a hardware library to the controller, right-click the Connected Libraries folder, select **Connect Library**, and select the hardware library.

Only hardware libraries that are added to the project can be connected to a controller (see Libraries Folder on page 29).

For each controller, there is a CPU unit to which other hardware units, such as I/O units and communication interfaces can be added. Units can also be added to the controller on the same level as the CPU unit. The controller structure mirrors the physical structure, which means that all ports and buses have their own corresponding unit (icon) in Project Explorer.

For more information about hardware configuration and the Controllers folder, see Section 4, Hardware Configuration.

The Controllers folder also contains a Tasks sub-folder and an Access Variables container. The Tasks folder contains tasks that are used to control the execution of your applications. By default, the Tasks folder contains three tasks: Fast, Normal, and Slow. However, you can add the tasks you need for your applications.

For more information on tasks and task execution, see the manual *AC 800M, Configuration*.

Double-clicking the Tasks folder will display a task overview. Double-clicking an individual task will display the Task Properties dialog for that particular task.

From objects in the Controllers folder (CPU units, I/O units, communication ports, communication interfaces, etc.), a number of hardware editors can be opened, see Editors on page 37.

Drag-and-Drop in Project Explorer

The Project Explorer supports drag-and-drop operations.

Dragging to Text Input Fields

All objects can be dragged to an arbitrary text input field or text editor. When the object is dropped, the current name of the object becomes the text input. This helps in deriving names for variables, parameters, and function blocks, from the existing object names.

For example, in Figure 9, the name of the variable is the result of a drag-and-drop operation from the library *FBCReactorLib* to the Name column in the Function Block editor. The text can be modified in the Name column.
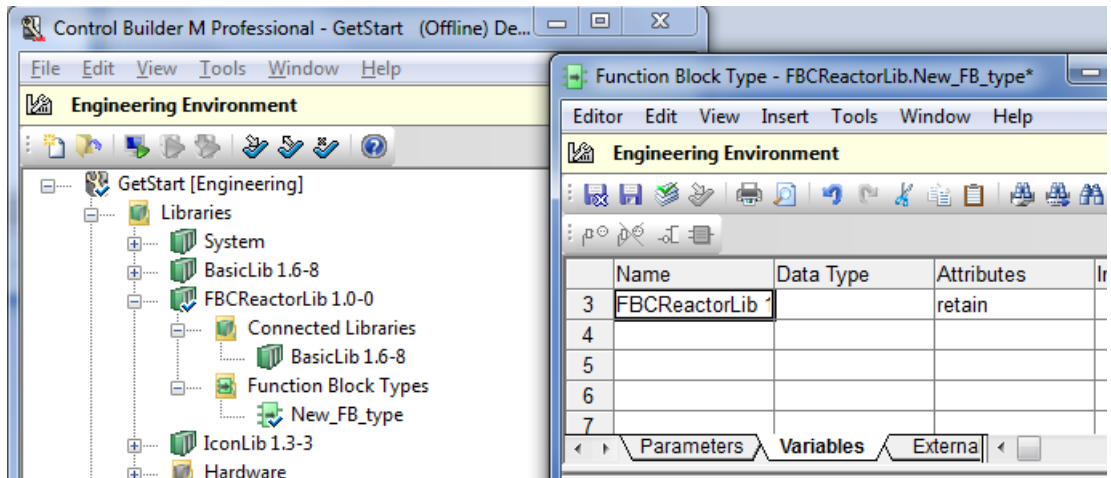

Figure 9. Parameter name derived by dragging to text input field

Dragging to Objects

Some objects can be dragged to other objects. Table 1 shows the supported actions.

Table 1. Drag-and-Drop operations with objects

Drag Source	Drop Target	Operation
Library	Application or another library	Connects the library to the application or the target library. The source library is then visible in the Connected Libraries folder in the target application or library.
Hardware Library	Controller	Connects the hardware library to the controller. The source hardware library is then visible in the Connected Libraries folder in the target controller.
Application	Controller	Assigns the application to the Controller. The source application is then visible in the Connected Applications folder in the target controller. **Note:** If the application is an application reference object (that is, an application shown below the "Assigned Applications" object for a Controller), then this is a Move operation that removes the previous assignment.
Application	Task	This results in two operations: • Assigns the application to the task. • Assigns the application to the corresponding controller. The source application is then visible in the Connected Applications folder in the corresponding controller.
Application	Application Folder	Moves the application to the target application folder.
Application Folder	Application Folder	Moves the application folder and its contents to the target application folder.

Context Menus

Context menus can be used to edit the properties of various objects. Context menus are displayed by right-clicking an object in Project Explorer.

Unless you have reserved an entity, parts of the Project Explorer will be read-only, for example, some context menu items will be disabled, and dialog boxes may be read-only.

Reservation Status

Entity icons can indicate reservation status, see Figure 5. For example:

*   indicates that an application is not reserved by anyone.
*   indicates that an application is reserved by you.
*   indicates that an application is reserved by another user.

Message Pane

See the location of the message pane in Figure 4. The message pane contains three tabs:

*   Description, shows a description of the selected type or hardware object.
*   Check, shows the result of a code check, including error messages.
*   Message, showing messages resulting from events in Control Builder, such as compiling and loading a new project.

Editors

Control Builder contains a number of editors. The editors can be accessed from Project Explorer and from Plant Explorer. To access an editor, right-click the object (it could be a controller, another hardware unit, an application, a program, a diagram or a type) and select the editor from the context menu.
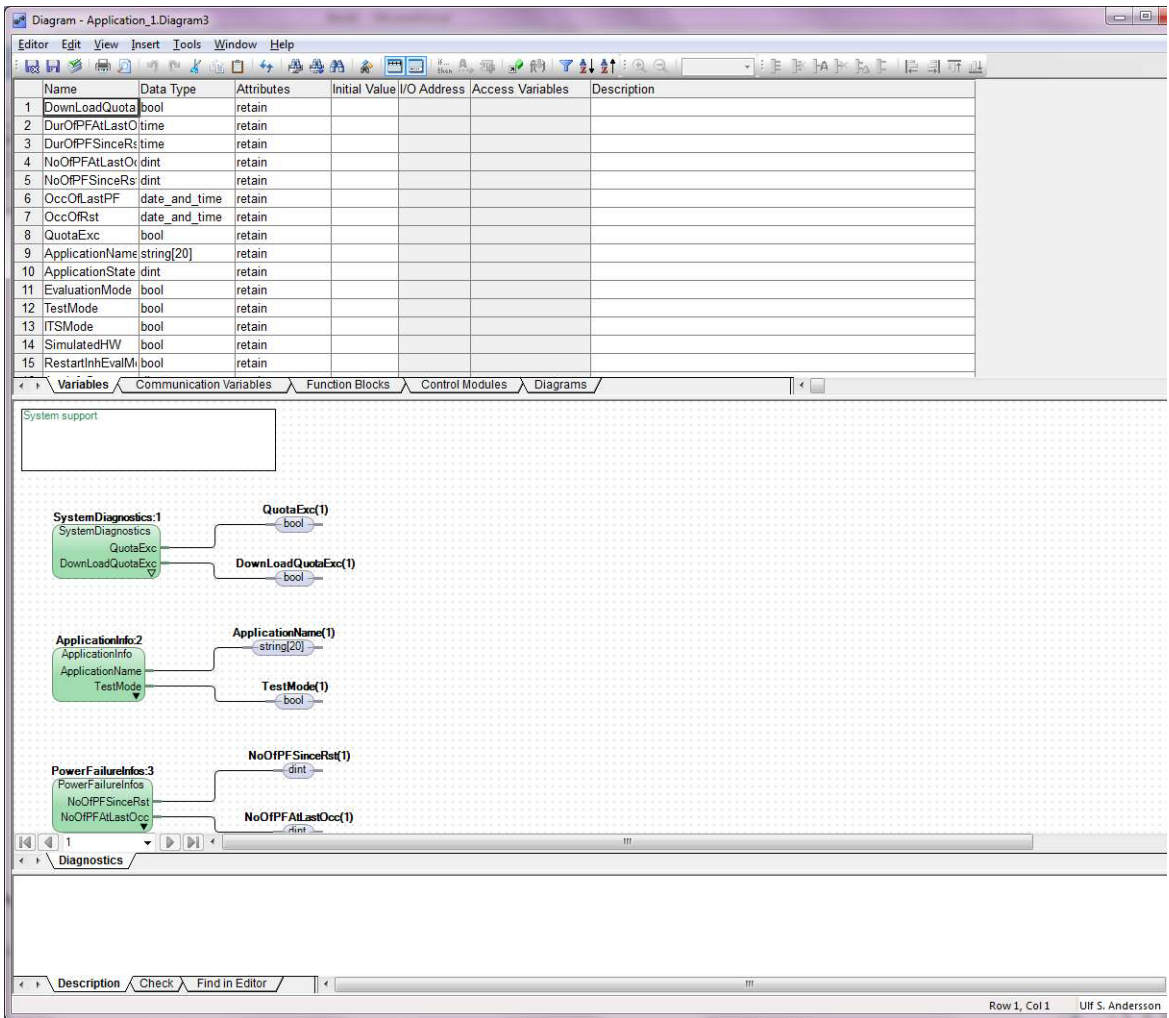
Figure 10. Diagram Editor

Among many things, editors are used to declare variables and connect parameters to I/O channels. There are also a number of programming language-specific editors,

such as the Function Block Diagram (FBD) editor and the Control Module Diagram (CMD) editor.

Refresh

The Refresh function loads the latest code from the aspect server and can be used on the following levels: project, library, application and controller.

Refresh a Project

To refresh a project:

1.   In Project Explorer (with your project open), right-click the project icon and select **Refresh Project** in the context menu. The Project will load information from your system.

Refresh a Library, Application or Controller

To refresh a library, application, or controller:

1.   In Project Explorer (with the control project active), expand **Library, Application** or **Controller** and right-click this object.

2.   Select **Refresh**.

Section 3 MyDoors Project

Introduction

This section helps you to build a small project and get yourself familiar with Control Builder. The guidelines that come with this example suggests that you build a project called MyDoors that simulates the entrance to a store. While working with the MyDoors project, you will learn about diagrams and declaration of variables and function blocks in diagrams, and much more.

Control Builder comes with a ready-made project example called ShopDoors installed on your local disk, see Ready-made Projects for CB Professional on page 95 for locating the ShopDoors example or any other Control Builder examples.

If you do not have access to an AC 800M controller or IO modules, you can still follow this example with a SoftController. Look for SoftController specific instructions throughout MyDoors project example.

The SoftController is test and debugging tool used for basic program tests. Execution of specific functions differs between SoftController and a hardware controller in several aspects, for example, execution performance, floating point precision (64 versus 32 bit), communication capabilities and so on.

This example will also show the additional steps necessary when you have two environments: Engineering and Production.

After finishing your study of the MyDoors project, you are advised to continue with the remaining sections in this manual.

Building a Shop Door Project

At the end of the MyDoors project, you will test your application in the Control Builder Test mode. By doing so, the Control Builder helps you to verify, in

a secure way, how variable values and conditions are changing during a diagram execution.

Before you create your project and start writing code, take a brief moment and study the given specifications and the suggested variable definitions below.

Specifications

This project will simulate the entrance to a store. The following specifications are given:

- The entrance consists of two sliding doors that open when a customer activates a photocell.

- Each door is opened and closed by its own motor.

- The doors return to default position (closed) five seconds after the last activation of the photocell. Consequently, several customers arriving one after the other will extend the time the door remains open.

- The number of customers is recorded for statistics. Manual reset of this counter should be possible.

- The total number of times the doors have opened since they were last serviced should be recorded.

- Each opening of the doors should increment a counter. When the counter reaches a preset limit, a flag should indicate that service is required. Manual reset of the flag should be possible.

Defined Variables

- **Photocell**
  The photocell has two states, active and inactive, typically represented by a Boolean variable. In this project, a Boolean variable named `Photo_Cell` (true = active, false = inactive) is used.

- **Door motors**
  The entrance itself consists of two doors facing each other. Each door is opened by a motor controlled by Boolean signals (`Motor_1` and `Motor_2`). The time the doors should remain open is declared in a variable `DoorsOpen_Time` of type Time.

- **Number of customers**
  Each time the photocell is activated, a counter representing the total number of customers entering the shop should be incremented. The counter, `Customers_Qty`, is of type Integer.

- **Reset the counter on certain dates**
  On certain dates, the shop manager records the total number of customers up to that date, and resets the counter. Consequently, a Boolean variable `Reset_Counter` is declared, which resets the counter.

- **Door service intervals**
  The doors should have regular service intervals, approximately after every 10,000 openings; you also need to keep a record of the number of openings from the previous service. The record is represented as the variable `Openings_Freq` of type Dint.

- **Time for service**
  When the counter reaches the upper limit defined by `Openings_Total` of type Dint, a flag (`Service_Req` of type Boolean) is set, indicating that service is required. This flag can be accessed by all controllers in the network. Manual reset of the service counter is activated using a Boolean variable `Serviced`. The doors should continue to work even if service is not performed.

Creating MyDoors Project

Starting the Control Builder

Double-click the Control Builder icon on the desktop (if selected during installation), or from the Start menu on the Windows Task Bar.

The Control Builder starts and open the Project Explorer interface.

1.   From the Project Explorer, select **File > New Project**, or click the 📄 icon. A New Project window opens.

       If you do not use Environments, ignore Step 2 to Step 4 below, and proceed with Step 5.

2.   Click the 🏭 icon to select the Environment.

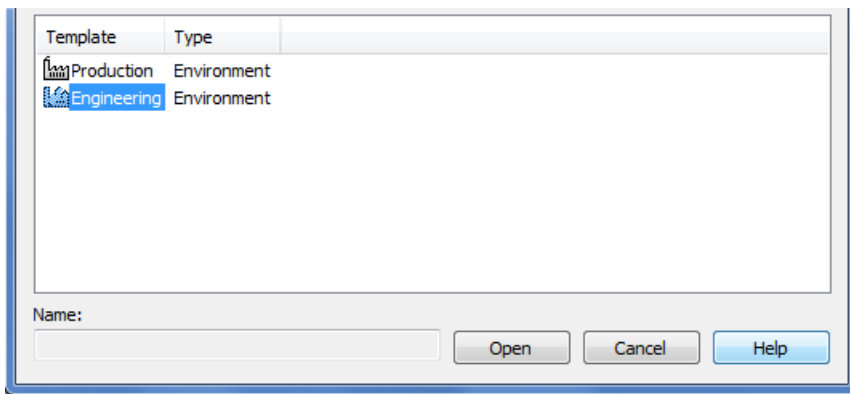3.   Select Engineering Environment from the list.

Figure 11. Selecting an environment.

4.   Click **Open**. The window shows available project templates.

5.   Select the **AC800M** template and type *MyDoors* in the **Name** field.
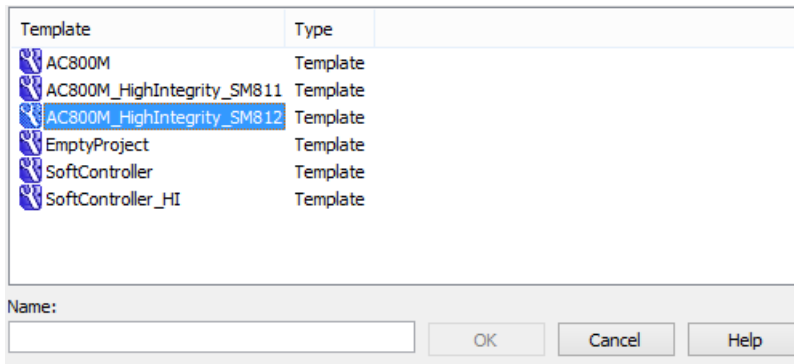
Figure 12. Standard templates for building a project.

6.   Click **OK**. Project Explorer creates and opens MyDoors project, see Figure 13.

Figure 13. MyDoors project opened in Project Explorer.

The Libraries folder contains the standard libraries Basic library (BasicLib), Icon library (IconLib) and Basic hardware library (BasicHWLib).

The System folder is always automatically inserted into a project. It contains firmware functions and cannot be removed from the project or changed by the user.

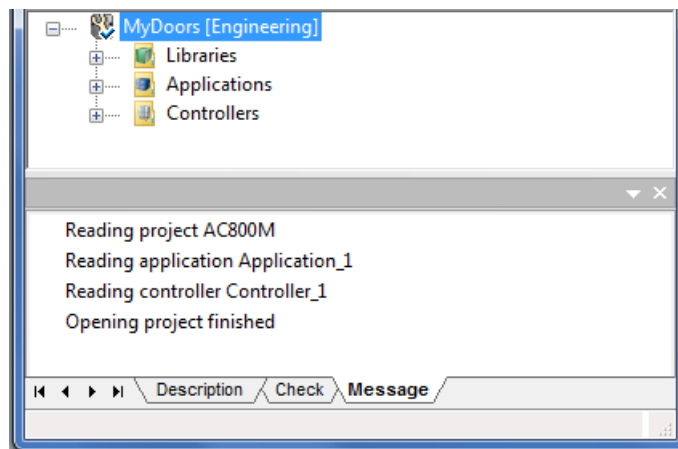> When you create a new project, the default application (Application_1) and the default controller (Controller_1) are automatically reserved.

Diagrams

In Control Builder, the control logic can be created in diagrams, which are created under an application. The diagrams use a graphical programming language.

The code block in the diagram editor allows mixing of functions, function blocks, control modules, and other diagrams, and allows graphical connection between them to achieve a particular logic.
In this example project, you are going to create the logic in a diagram.

In the MyDoors project, expand Application_1 to view the Diagrams folder and the three default diagrams.

In this example, you are going to create the logic in **Diagram2**.

Variables

There are different types of variables in Control Builder for storing and computing values (local, communication, global, and access variables).

The local variables are the most frequently used in Control Builder. As their name applies, the local variables belongs to the local code inside a function block, control module, diagram or program.

Communication variables are used to communicate between applications in the same controller or between different controllers in the network. The name of the communication variable must be unique within the network. Within an aspect directory, Control Builder automatically finds the referenced communication variables. If the communication variable is accessed from another aspect directory then the IP address needs to be specified.

In this example, you will declare 10 local variables and one communication variable in the editor for Diagram2.

Declaring Local Variables and the Communication Variable

1. Double-click **Diagram2** to open the corresponding Diagram editor.

2. The Diagram editor is divided into three panes: the declaration pane, the code pane, and the message pane. (See Figure 14).
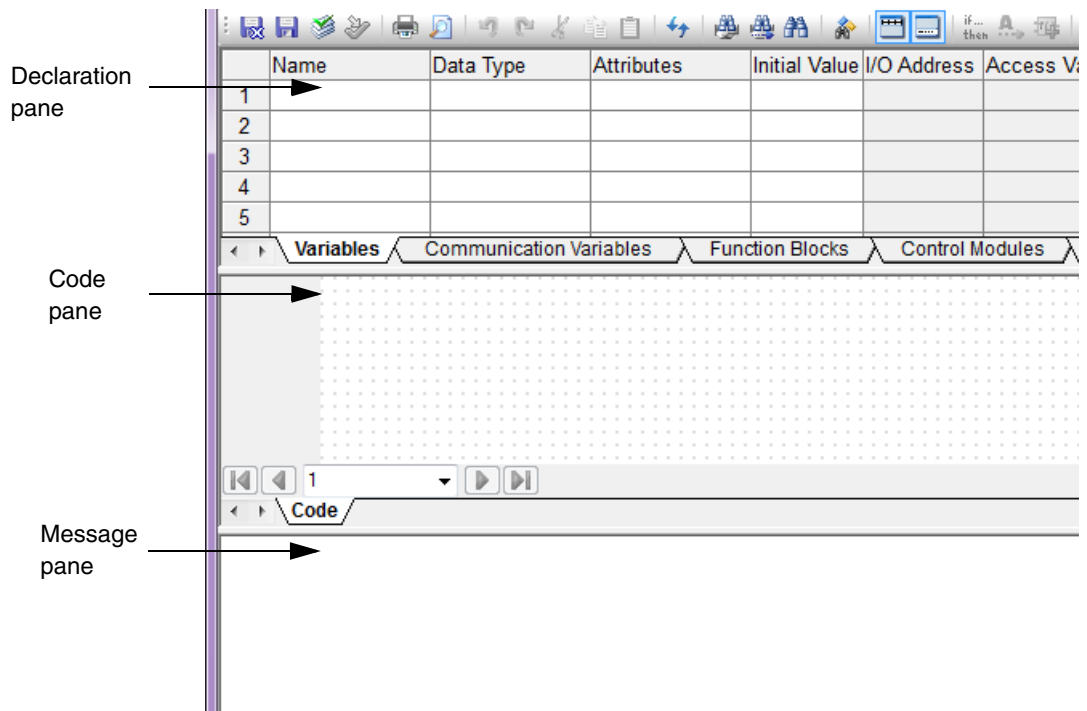
Declaration
pane

Code
pane

Message
pane

Figure 14. The editor for Diagram2.

3. Select the **Variables** tab, and place the cursor in the upper left-hand cell in the declaration pane and type `Photo_Cell`.

4. Move one cell to the right by pressing the tab key. Type `bool` in the "Data type" column. Move the cursor to next column labeled "Attributes".

5. Choose the default setting `retain` (which means that the variable will keep its value at a warm restart). Press the tab key to move to the next column.

6. Set the initial value to `false` to indicate that the doors are closed at start-up.

7. Skip the column I/O address. The address will be automatically filled in later when connecting variables to I/O channels.

8. The last column 'Description' is reserved for you to use freely. Your first row should now look like row 1 in Figure 15.

	Name	Data Type	Attributes	Initial Value	I/O Address	Access Variables	Description
1	Photo_Cell	bool	retain	false			Photo cell to doors

Figure 15. Declaration of the Boolean variable Photo_Cell

9. Declare a second variable named, `DoorsOpen_Time` which represents how long (duration time) the doors should remain open. Complete the declaration of `DoorsOpen_Time` according to row 2 in Figure 16.

Name	Data Type	Attributes	Initial Value	I/O Address	Access Variables	Description
Photo_Cell	bool	retain	false			Photo cell to doors
DoorsOpen_Time	time	constant	T#5s			Time duration that doors should be opened

Figure 16. Declaration of the DoorsOpen_Time variable

Note the attribute `constant` of the variable `DoorsOpen_Time`. You can either explicitly type "constant", or scroll through the available formats using Alt-key together with the up and down arrow keys, or press CTRL+J to display the list of attributes and then select `constant`.

10. Declare the next variable `DoorsOpen_ET` using a different method. Right click the code pane (grid area), and select **New > Variable**. In the New Variable dialog, declare the values for the variable as shown in Figure 17 and click **OK**.

Figure 17. New Variable dialog

The new variable, `DoorsOpen_ET`, gets inserted as a graphical object in the code pane. It also appears in the Declaration Pane as row 3.
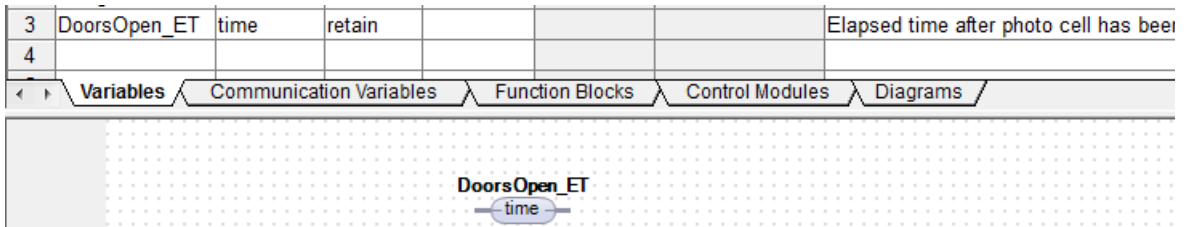
| 3 | DoorsOpen_ET | time | retain | | | | Elapsed time after photo cell has beer |
| 4 | | | | | | | |

| Variables | Communication Variables | Function Blocks | Control Modules | Diagrams |

DoorsOpen_ET
— time —

Figure 18. Declaration of the DoorsOpen_ET variable

11. Declare the remaining variables using any of the two methods (starting from row 4 to row 10) in the grid according to Figure 19.

3	DoorsOpen_ET	time	retain				Elapsed time after photo cell has been activated
4	Motor_1	bool	retain	false			Output to motor opening door 1
5	Motor_2	bool	retain	false			Output to motor opening door 2
6	Openings_Total	dint	constant	10			Total number of openings until service
7	Openings_Freq	dint	retain				Number of openings since last service
8	Serviced	bool	retain	false			Flag that resets the number of openings
9	Customers_Qty	dint	retain	0			Total number of customers
10	Reset_Counter	bool	retain	false			Flag that resets the number of customers
11							

Figure 19. Declaration of the remaining variables.

12. Select the **Communication Variables** tab in the Diagram editor, and declare the `Service_Req` communication variable as shown in Figure 20.

	Name	Data Type	Attributes	Direction	Initial Value	ISP Value	Acknowledge Group	Interval Time	IP Address	Description
1	Service_Req	bool	retain	out	false		Auto	normal	auto	Flag that is set when service is required
2										
3										

| Variables | **Communication Variables** | Function Blocks | Control Modules | Diagrams |

Figure 20. Declaration of the communication variable Service_Req

13. Click **Check** to check for errors.

14. Click **Save** to save the variables.

Door Timer and Customer Counter with Function Blocks

Timers and counters in Control Builder are normally represented as function block types and located in the Basic library. This example will declare one Timer (TOf), and two Counters (CTU) from the Basic library.

Declaring Function Blocks using Diagram Editor

Make sure the diagram editor for Diagram2 is open. The default code block (named Code) is the FD (Function Diagram) code block, which is a graphical editor (with visible grids) for adding graphical objects.

The code block – Code – consists of a default page, named 1.

To start creating the logic in the editor for Diagram2:

1. Rename the default page as **Motors_Doors**:

    a. Click the list box at the bottom of the code pane, and highlight the current page name, 1.

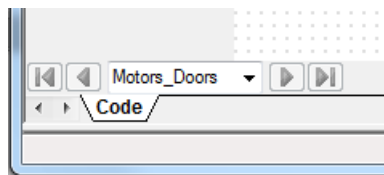   b. Replace 1 with the new page name, *Motors_Doors*.

Figure 21. Page 1 renamed as Motors_Doors

2. In the grid area of the Motors_Doors page, right-click to open the context
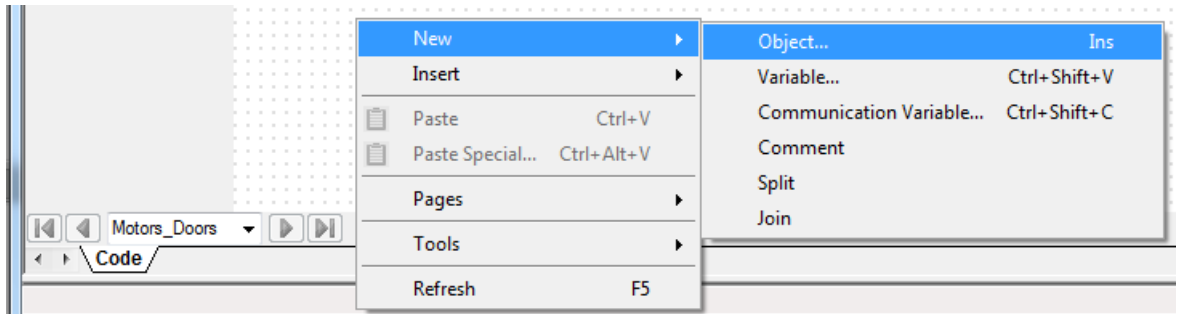   menu, and select **New > Object**. The **New Object** dialog opens.

Figure 22. Opening the New Object dialog

3. In the New Object dialog, select the **TOf** function block type from any of the
    two tabs:

    – The List tab contains alphabetical list of all objects available in the
      inserted libraries in Project Explorer. Select **TOf**.

    – The Tree tab displays a tree structure to navigate to the required object.
      Go to **Connected Libraries > BasicLib > Function Block Types**, and
      select **TOf**.

4. After selecting TOf, enter **OpenDoors** in the Name field as shown in
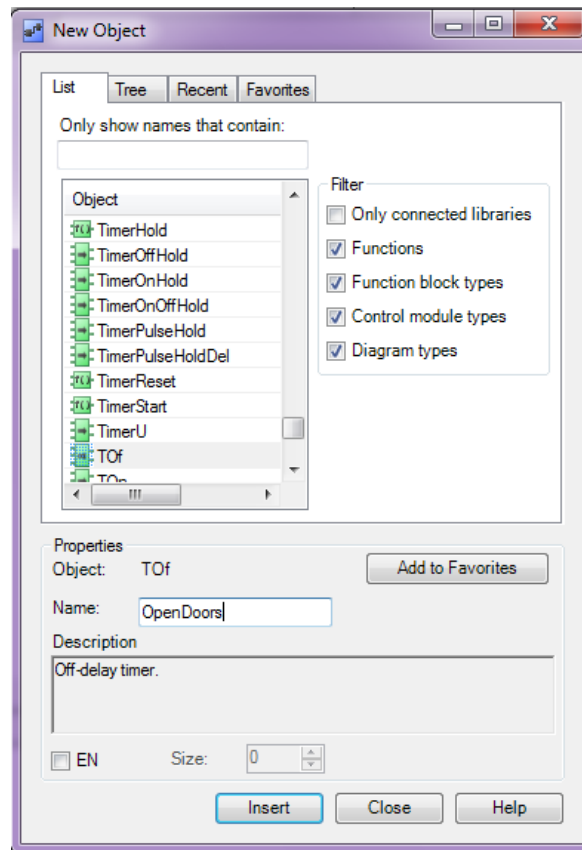   Figure 23.

Figure 23. New Object dialog

5. Click **Insert**, and then click **Close** to close the dialog.

    OpenDoors function block is inserted as in the Figure 24.

6. In the declaration pane for Function Blocks, add the description 'Timer for Motor' corresponding to the OpenDoors row.

	Name	Function Block Type	Task Connection	Description
1	OpenDoors	BasicLib.TOf		Timer for motor
2				
3				
4				
5				

◄ ► \ Variables \ Communication Variables \ **Function Blocks** \ Control Modules \

OpenDoors:1
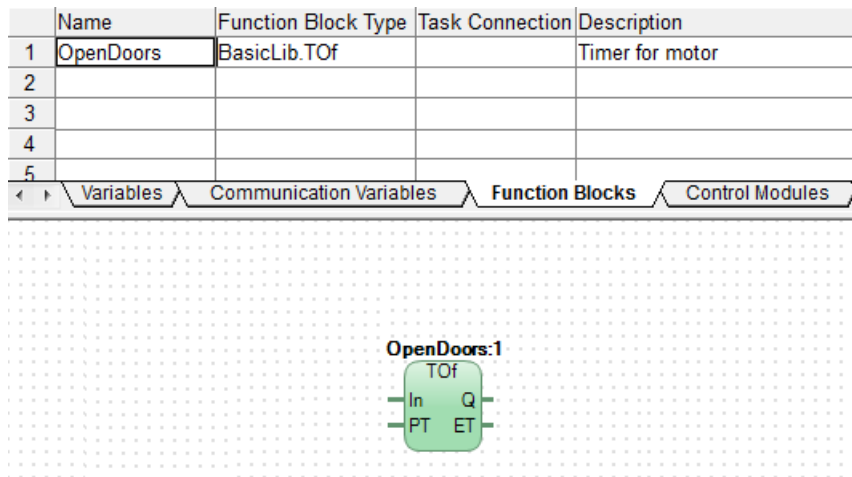TOf
In   Q
PT   ET

Figure 24. Declare TOf Function Block Type in Motors_Doors page

7. Add a page, by selecting **Insert > New Page** or, right-click on the grid area and select **Pages > Add New Page**. Rename the new page as **Number_of_customers**.

8. Open the **Number_of_customers** page in the diagram editor. In the grid area of code pane, right-click and select **New > Object** from the context menu. The **New Object** dialog opens.

9. Select **CTU** from the list and enter **Customer_Count_Up** in the Name field. Click **Insert** and close the dialog.

10. Customer_Count_Up function block is created as in the Figure 25. Add the description according to Figure 25.

	Name	Function Block Type	Task Connection	Description
1	OpenDoors	BasicLib.TOf		Timer for motor
2	Customer_Count_Up	BasicLib.CTU		Counter for the number of customers
3				
4				
5				

◄ ► \ Variables \ Communication Variables \ **Function Blocks** \ Control Modules \ Diagrams /
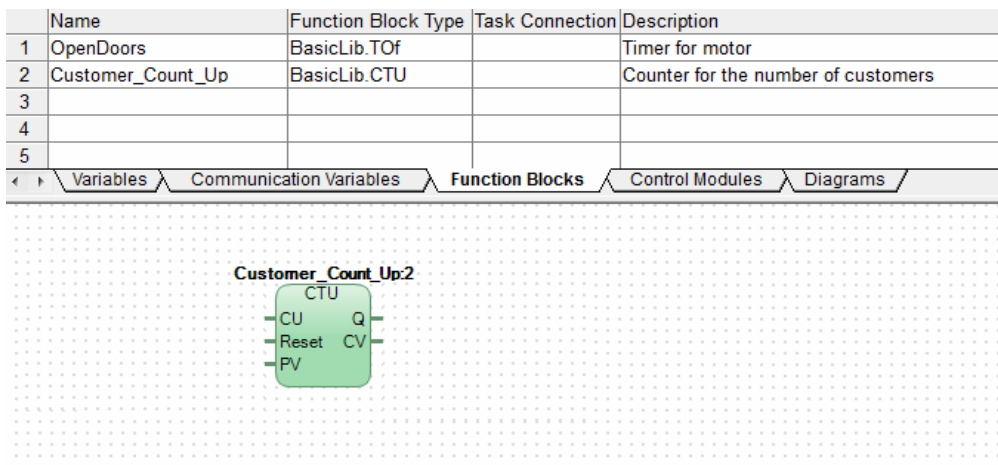
Customer_Count_Up:2
CTU
CU   Q
Reset   CV
PV

Figure 25. Declare CTU Function Block Type in Number_of_Customers page

11. Add a third page, by selecting **Insert > New Page** or, right-click on the grid area and select **Pages > Add New Page**. Rename the new page as **Service_On_Doors**.

12. Open the **Service_On_Doors** page in the diagram editor. In the grid area of code pane, right-click and select **New > Object** from the context menu. The **New Object** dialog opens.

13. Select **CTU** from the list and enter **Service_Count_Doors** in the Name field. Click **Insert** and close the dialog.

14. **Service_Count_Doors** function block is created as in the Figure 26. Add the description according to Figure 26.

	Name	Function Block Type	Task Connection	Description
1	OpenDoors	BasicLib.TOf		Timer for motor
2	Customer_Count_Up	BasicLib.CTU		Counter for the number of customers
3	Service_Count_Doors	BasicLib.CTU		Counter for the number of openings of doors
4				
5				

◄ ► \ Variables ⋀ Communication Variables ⋀ **Function Blocks** ⋀ Control Modules ⋀ Diagrams /

```
          Service_Count_Doors:3
                 CTU
            CU        Q
            Reset     CV
            PV
```

Figure 26. Declare CTU Function Block Type in Service_On_Doors page

For more information about the TOf and the CTU function blocks, open the Control Builder Online Help. Simply place the cursor in the Function Block Type cell (for example TOf), and press **F1**.

15. Click **Check** ✓ to check for errors.

Creation of Logic

The creation of logic in the diagram editor involves inserting and connecting the graphical objects, which can be also be moved freely in the grid area.

Making a Function Block Call in Motors_Doors

1. Open the **Motors_Doors** page. In the grid area right-click and select **Insert > Variable**. From the variable list, select **Photo_Cell** and click **Insert**.

2. Drag one port of **Photo_Cell** and connect to **In** parameter of the **OpenDoors** function block graphically.

3. The variables can also be inserted by drag and drop method. From the Variables tab, click the **DoorsOpen_Time** variable and drag it to the grid area.

4. Insert the variables **Motor_1** and **Motor_2** using any of the methods. The variable **DoorsOpen_ET** is already present in the grid area. Connect these three variables to the parameters as in Figure 27.
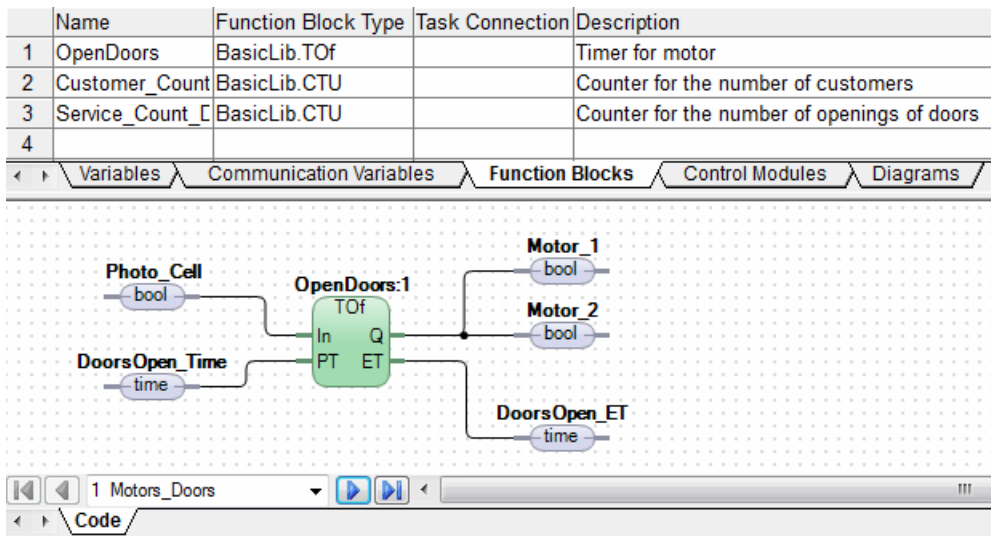

Figure 27. Variable connections to OpenDoors Function Block

The variables can also be connected by right-clicking the port and selecting **Connect**. In the Connect dialog, enter the first few characters of the variable to select the variable from the list. Click **OK**.
If the variable is not declared earlier, after entering the variable name in the Connect dialog, a **Declare Variable** dialog is shown. Select **Variable** or **Communication Variable**, and click **Declare**.

5. Click **Save and Close** .

What About the Q parameter in TOf?

The output Q parameter is a Boolean signal, which represents the status on the door position (open or closed) and is passed on to the motors. For both doors to open, the Q signal must be passed to both motors. The variables Motor_1 and Motor_2 are connected as in the Figure 27.

Making a Function Block Call in Number_Of_Customers

1. Open the **Number_Of_Customers** page. In the grid area, insert the variables **Photo_Cell**, **Reset_Counter**, and **Customers_Qty**.

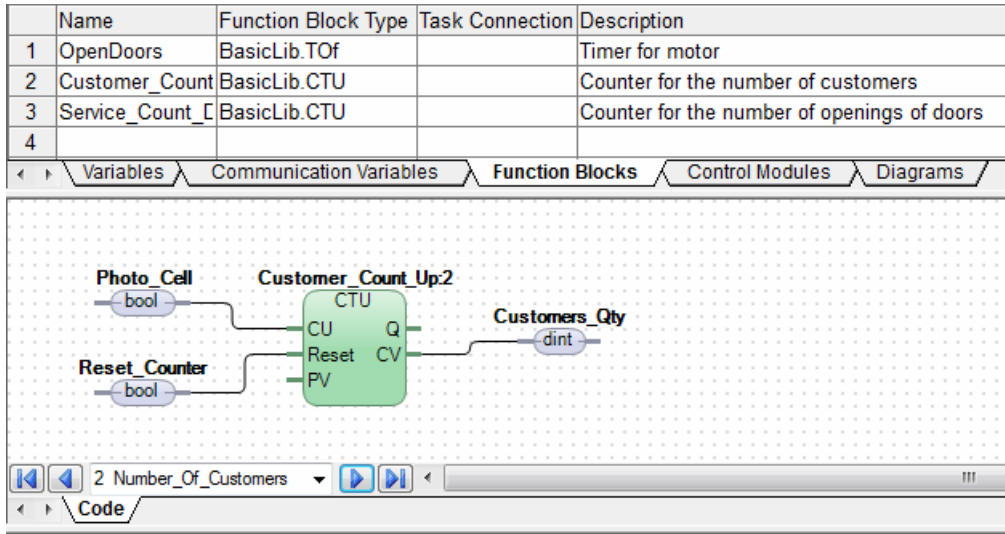2. Connect the variables to the **Customer_Count_Up** Function Block type as in the Figure 28.

	Name	Function Block Type	Task Connection	Description
1	OpenDoors	BasicLib.TOf		Timer for motor
2	Customer_Count	BasicLib.CTU		Counter for the number of customers
3	Service_Count_D	BasicLib.CTU		Counter for the number of openings of doors
4				

◄ ► \ Variables \ Communication Variables \ **Function Blocks** \ Control Modules \ Diagrams /

Figure 28. .Variable connections to Customer_Count_Up Function Block Type

3.   Click **Save and Close** .

Making a Function Block Call in Service_On_Doors

1.   Open the **Service_On_Doors** page. In the grid area, insert the variables **Motor_l**, **Serviced**, **Openings_Total**, **Service_Req**, and **Openings_Freq**.

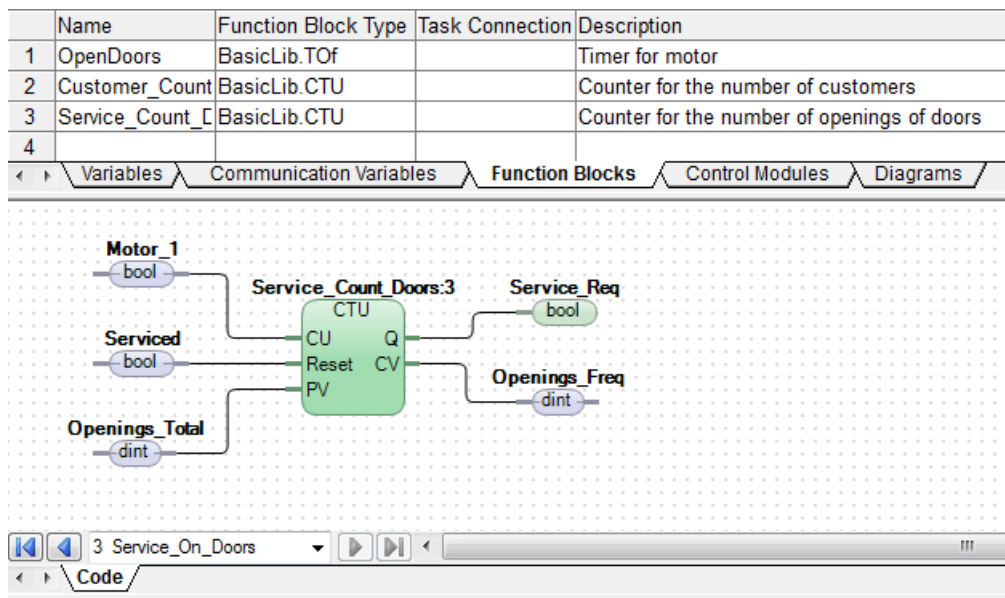2.   Connect the variables to the **Service_Count_Doors** as in the Figure 29.

	Name	Function Block Type	Task Connection	Description
1	OpenDoors	BasicLib.TOf		Timer for motor
2	Customer_Count	BasicLib.CTU		Counter for the number of customers
3	Service_Count_D	BasicLib.CTU		Counter for the number of openings of doors
4				

Variables \ Communication Variables \ **Function Blocks** \ Control Modules \ Diagrams

Figure 29. Variable connections to Service_Count_Doors Function Block Type

3. Click **Save and Close** .

> If an error message should be displayed in the message pane, double-click
> the error line and you will jump directly to the error location in the code.
> You will also find a brief description in the message pane, explaining the
> type of error that has occurred.

Testing MyDoors Project

Before downloading the application to a controller and going online, it is often
necessary to first test the application in an offline mode and confirm that everything
is working properly. This mode is called the Test Mode and means basically that
Control Builder will compile and execute the code locally in the PC as if it was
downloaded to an AC 800M controller.

The test mode is an easy way to try out the application many times. However, external communication will be disabled during the test mode, thus reading and writing variables connected to IO units cannot be validated in test mode.

The communication variable cannot be tested in Test Mode.

Before running the diagram in Test mode, there is an option to enable the Difference Report window. However, the Difference Report function is not important for this example since it does not generate a report in Test mode. For details on how to enable this function, see Difference Report on page 107. This example assumes that the Difference Report has the default setting (not enabled).

1. In Project Explorer, click **Test Mode** . The Test Mode Analysis window opens.

2. Click **Cold Restart All**.

3. Click **Continue**.

4. Double-click **Diagram2** to display the editor.

5. Open **Motors_Doors** page. All variables in **Diagram2** are listed in the upper pane and the diagram appears in the lower pane, see Figure 30.

Name	Current Value	Data Type	Attributes	Initial Value	I/O Ad
Photo_Cell	false	bool	retain	false	
DoorsOpen_Time	0d0h0m5s0ms	time	retain	T#5s	
DoorsOpen_ET	0d0h0m0s0ms	time	retain		
Motor_1	false	bool	retain	false	
Motor_2	false	bool	retain	false	
Openings_Total	10	dint	constant	10	
Openings_Freq	0	dint	retain		
Serviced	false	bool	retain	false	
Customers_Qty	0	dint	retain	0	
Reset_Counter	false	bool	retain	false	

◄ ► \ **Variables** / Communication Variables / Function Blocks / Control Modules / Diagrams /

Figure 30. The Diagram editor in Test mode.

Analyzing the Code During Diagram Executions

As you can see, test mode helps you test and analyze your project without yet having any hardware configured in the Project Explorer tree. You can change the variable values and study the diagram response.

While analyzing the variable conditions, the following instructions will ask you to right-click a variable and change its value. The variable values can be right-clicked from either the parameter list or directly in the code pane.
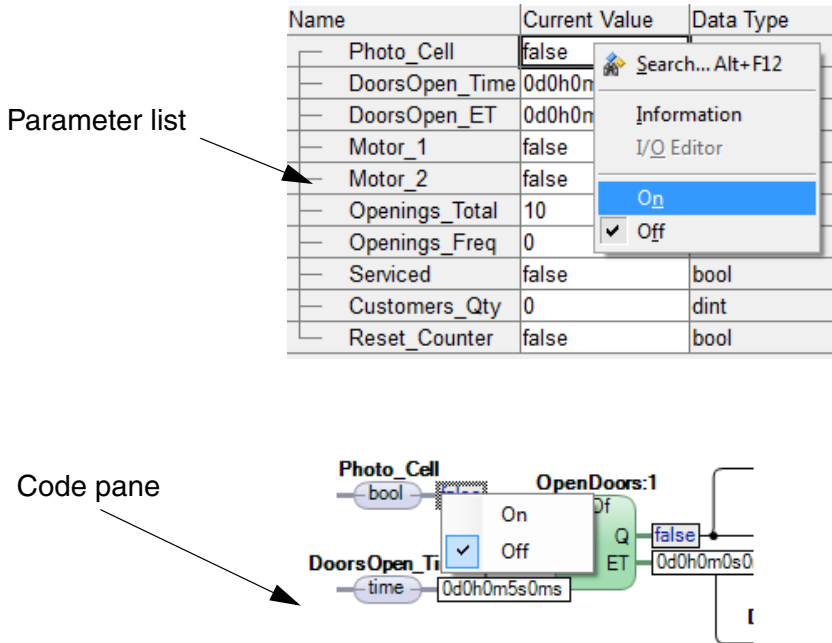
Parameter list

Code pane

Figure 31. Changing the current value on a variable from the Parameter list, or in the code pane.

1. Right-click `Photo_Cell` and select **On** in the context menu.

   Note that the motors change to True (start) and the number of openings since last service increases by one, as does the number of customers.

2. Right-click `Photo_Cell` and select **Off** in context menu.

   Simulating that no customer is activating the photocell. Note how the clock starts and counts up to five seconds at which point the motors are set to False (stop) and the doors close.

3. Right-click `Photo_Cell` and select **On**, then QUICKLY select **Off** again.

   Simulating that a customer has activated the photocell. Both the number of openings is increased and customers increase.

4. Wait until the doors close. Right-click `Photo_Cell` and QUICKLY select **On**, **Off**, **On**, **Off**, **On**, **Off.**

    Simulating that three customers are passing the photocell one by one. Notice that the clock starts when the first customer passes the photocell and resets to 0 when the next customer passes. Consequently, the opening time is extended for a new period of 5 seconds, and so on. Note also that the number of times the doors open only increases by one, whereas the number of customers is increased by three. You should have three openings of the doors and five customers registered.

5. In the variables list, right-click `Reset_Counter` and select **On**, then select **Off** again. Reset the customer counter.

6. Activate the photocell so the number of openings (`Openings_Freq`) passes `Openings_Total`. `Service_Req` will then become *True*.

7. Right-click `Serviced` and select **On**, then select **Off** again.

    Study the reaction of the counters and flags. Note that the variable `Openings_Freq` resets.

8. Close Diagram editor.

9. From Control Builder Menu bar, select **Tools > Stop Test Mode.**

Section 4 Hardware Configuration

This section teaches you how to add or remove hardware units from the tree structure in the Project Explorer. It covers the necessary steps for building a software model that represents a limited part of a hardware configuration in the plant.

Configure Hardware

Study the hardware configuration in Figure 32. Assume an AC 800M controller, together with six I/O modules. We are going to add two of them (DO814 and DI810) to the tree structure in Project Explorer. The modules are placed at positions 1 respective 2.
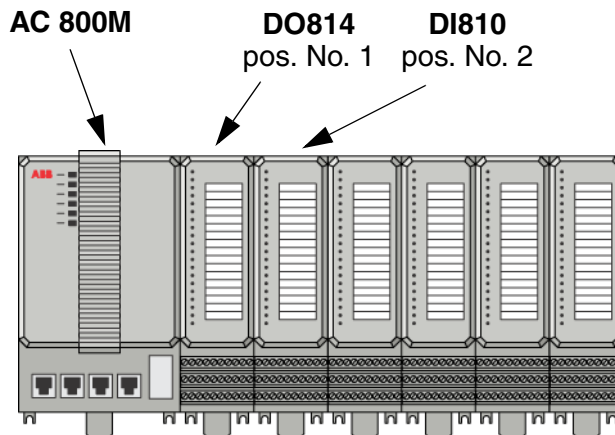
Figure 32. Hardware position for IO modules (for example DO814 at position 1 and DI810 at position 2).

Changing a CPU Unit

You must have the same CPU unit connected in Project Explorer as in the physical controller, otherwise you cannot download your application to the Controller. However, if you run with a SoftController with hardware simulation, the choice of CPU models is optional.

ℹ️   In this example, a default CPU PM860 will be replaced with a CPU PM864.

To replace a CPU:

1. Expand **Controllers > Controller_1 > Hardware AC 800M** until you see the **PM860 / TP830** item in the Project Explorer tree.

2. Right-click the **PM860 / TP830** item and select **Replace Unit** in the context menu. A 'Replace' window opens.

3. Expand **Connected Libraries** and select, for example **PM864/TP830**.
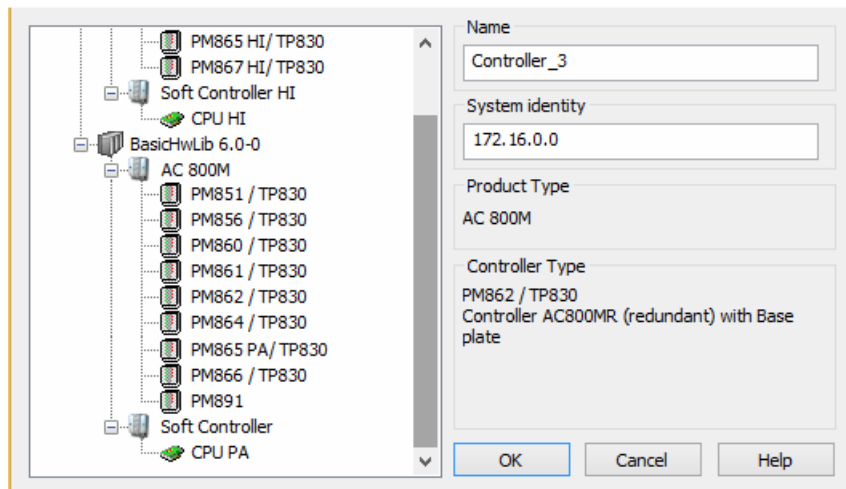
4. Click **Replace** and then **Yes** to accept the change.

Adding the IO Modules DO814 and DI810

The S800 IO modules are represented in Control Builder as hardware types located in the hardware library **S800IOModulebusHwLib**. Thus, before adding the IO

modules you must first insert the hardware library to your project. Once the library has been inserted to your project you can connect it to your hardware configuration and then access the IO modules and add them to your controller configuration.

To insert and connect a hardware library:

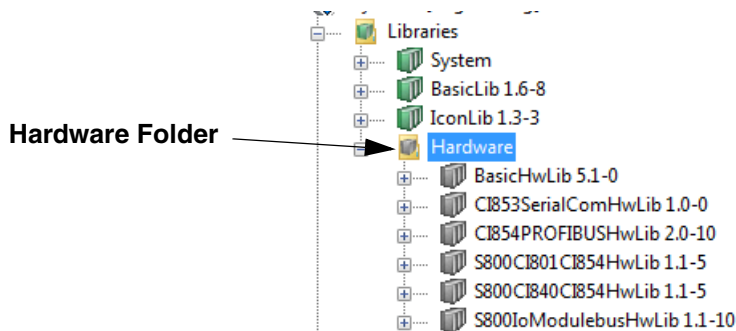1. Expand **Libraries** folder, until you see **Hardware** folder in the Project Explorer tree.

Figure 33. The hardware folder located inside Libraries folder in the Project Explorer.

Among the hardware libraries listed under the Hardware folder, the S800IoModulebusHwLib library contains S800 IO units for the Modulebus.

2. Expand **Controllers > Controller_1** until you see the **Connected Libraries** folder in the Project Explorer tree.

3. Right-click the **Connected Libraries** folder, select **Connect Library**, and select **S800IoModulebusHwLib** from the window.
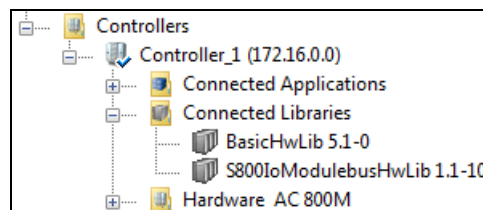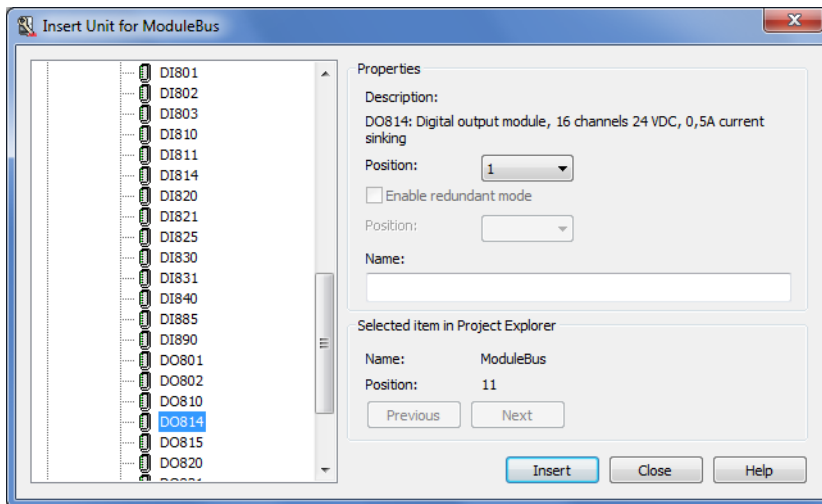
4. Click **OK**.

Figure 34. The new S800IO library has been connected to the controller.

Adding the IO modules from the hardware library:

1. Expand **Controllers > Controller_1 > Hardware AC 800M > PM864/TP830** until you see the **ModuleBus** item in the Project Explorer tree.

2. Right-click the **ModuleBus** item and select **Insert Unit** in the context menu. A 'Insert Unit for ModuleBus' window opens.

3. Expand **Connected Libraries > S800Io ModulebusHwLib > Hardware types** and select DO814.

4. Keep default position **1** from *Position* drop-down menu and click **Insert**.

5. Scroll up-down in the list and select **DI810.**

6. Keep default position **2** from *Position* drop-down menu and click **OK**.

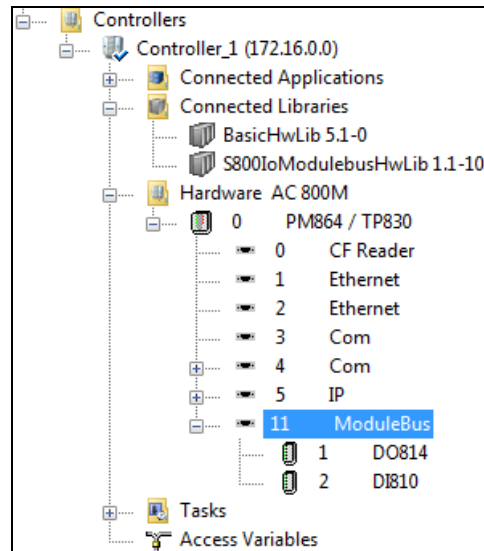   When you have added the two IO modules, your "hardware tree" should look like the configuration in Figure 35.

Figure 35. Hardware setup (Project Explorer).

> To remove a hardware unit, right-click the object in the tree structure and select Delete.

Connect Variables to I/O Channels

Communication between I/O channels and code is established by connecting variables to I/O channels. Therefore, Control Builder provides you with two different connection methods of choice. If this is your first occasion with Control Builder, you are advised to try both methods for learning. The previous MyDoors project will not be completed unless you follow both methods.

Method 1, will connect the variable Photo_Cell to the IO module DI810 by using dot notation. Method 2 - Using a Path Selector on page 70, will connect the variables Motor_1 and Motor_2, to the IO module DO814 via a path selector menu.

Method 1 - Using Dot Notation

Connecting the Photo cell to a DI810 Channel

Under the Controllers in Project Explorer:

1. Double-click **DI810** I/O module. The DI810 hardware editor opens.

2. Select the **Connections** tab and place the cursor in the first empty white cell.

3. Type **A** (for Application_1) and observe how the editor fills in the rest.

4. Press "." (dot) to move to the next level. All three Programs will be displayed.

5. Select **Diagram2**, and press "." (dot) again. A list of variables will open.

6. Select `Photo_Cell` in the list.

7. Press the ENTER key. The `Photo_Cell` variable has been connected to the first channel in DI810.

Channel	Name	Type	Variable	I/O Descrip
IX0.11.2.1	Input 1	BoolIO	Application_1.Diagram2.Photo_Cell	
IX0.11.2.2	Input 2	BoolIO		
IX0.11.2.3	Input 3	BoolIO		

Figure 36. The variable Photo_Cell connected to first IO Channel.

Method 2 - Using a Path Selector

Connecting the Motors to DO814 Channels

From the Controllers in Project Explorer:

1. Double-click the **DO814** I/O module. The hardware editor for DO814 opens.

2. Select the **Connections** tab and place the cursor in the first empty white cell (Channel QX0.11.1.1 in the **Variable** column).

3. Right-click **Insert > Insert Path From Tree** from the context menu**.** A list box opens.

The Menu bar action **Insert > Insert Path From Tree** can also be done by clicking the 🖳 icon located in the hardware editor, or by pressing CTRL+T inside the cell.

4.   Expand **Application_1 > Diagram2.** Double-click **Motor_1** to insert the full path, see Figure 37.

Channel	Name	Type	Variable	I/O Des
QX0.11.1.1	Output 1	BoolIO		
QX0.11.1.2	Output 2	BoolIO	⊟ Application_1	
QX0.11.1.3	Output 3	BoolIO	⊞ Diagram1	
QX0.11.1.4	Output 4	BoolIO	⊟ Diagram2	
QX0.11.1.5	Output 5	BoolIO	Customers_Qty	
QX0.11.1.6	Output 6	BoolIO	DoorsOpen_ET	
QX0.11.1.7	Output 7	BoolIO	DoorsOpen_Time	
QX0.11.1.8	Output 8	BoolIO	Motor_1	
QX0.11.1.9	Output 9	BoolIO	Motor_2	
QX0.11.1.10	Output 10	BoolIO	Openings_Freq	
QX0.11.1.11	Output 11	BoolIO	Openings_Total	
QX0.11.1.12	Output 12	BoolIO	Photo_Cell	
			Reset_Counter	
			Serviced	

Figure 37. The path for Motor_1 variable selected in the tree.

5.   Place the cursor in the second empty white cell (Channel QX0.11.1.2 in the **Variable** column) and connect Motor_2, by yourself. Use the 🔧 icon.

After connecting the variables, your hardware editor should look like the editor illustrated in Figure 38.

Channel	Name	Type	Variable	I/O Desc
QX0.11.1.1	Output 1	BoolIO	Application_1.Diagram2.Motor_1	
QX0.11.1.2	Output 2	BoolIO	Application_1.Diagram2.Motor_2	
QX0.11.1.3	Output 3	BoolIO		

Figure 38. The Motor_1 and Motor_2 connected to DO814.

6.   Click **Check** 🗹 for errors.

7.   Click **Save and Close** 💾.

Reading I/O addresses from the Application

An easy way to read the I/O address is to open (in this case) **Diagram2** in the program editor and check the column labeled **I/O Address**. Here you will find the address for the photocell and the motors (see Figure 39).

DI810

	Name	Data Type	Attributes	Initial Value	I/O Address
1	Photo_Cell	bool	retain	false	Controller_1.0.11.2.1
2	DoorsOpen_Time	time	constant	T#5s	
3	DoorsOpen_ET	time	retain		
4	Motor_1	bool	retain	false	Controller_1.0.11.1.1
5	Motor_2	bool	retain	false	Controller_1.0.11.1.2
6	Openings_Total	dint	constant	10	

DO814

Figure 39. The I/O Address column shows how variables are connected to I/O channels.

Changes made to I/O connections in the hardware editor will be reflected in both editors.

Your project has now been tested offline and the hardware configuration is complete.

Releasing Reservations

Since you have finished the configuration, you can release any reservations.

1. Right-click **MyDoors**, and select **Release Reservation**.

2. The displayed dialog box is used to release reserved entities. In this example, the default selection is sufficient. However, you can enter a comment.
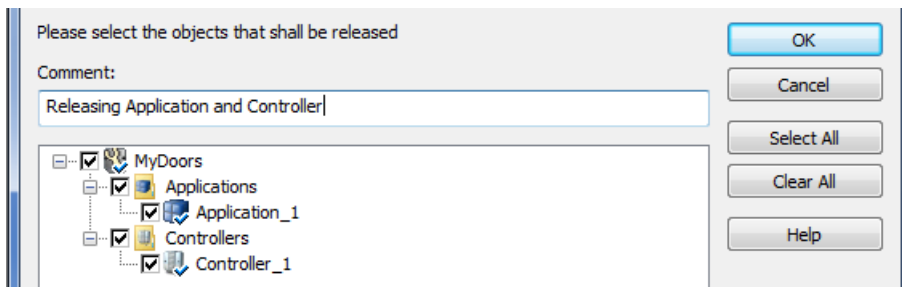

Figure 40. Releasing reservations.

3.   Click **OK**. The reservations are released.

If you use environments, continue by deploying the configuration (described below). Otherwise, you are ready to proceed with Section 5, Connecting the Controller and Go Online.

Deploying Configuration Changes

i   This section is only valid when using environments.

When you have released the reservations, you deploy the changes from the Engineering Environment to the Production Environment.

1.   Right-click **MyDoors** project, and select **Deploy**.

The displayed dialog box is used to select relevant destination environment, and what entities to deploy.
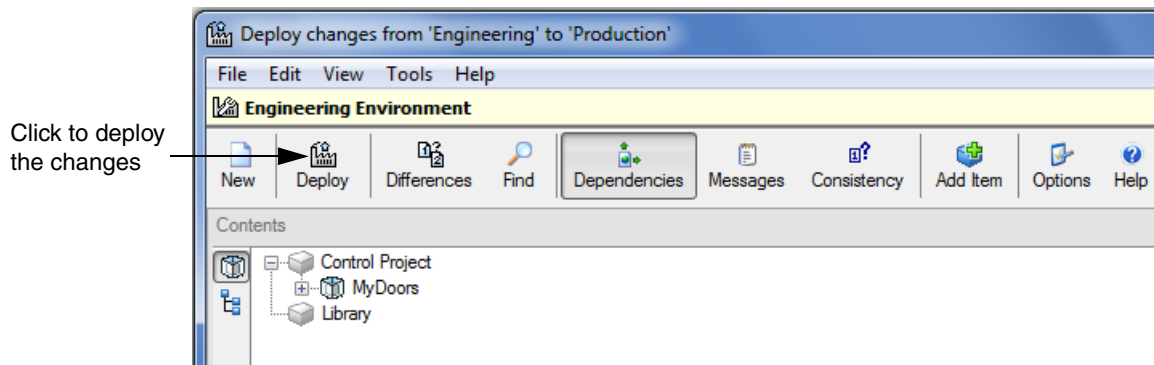
Figure 41. Deploy changes.

2.   Click **Deploy**. The project and all other modified entities are deployed to the Production Environment.

You are now ready to proceed with Section 5, Connecting the Controller and Go Online.

Section 5 Connecting the Controller and Go Online

This section contains the prerequisites for connecting a controller and the general procedure for downloading a project to the controller.

If you have created a project according to Section 3, MyDoors Project and then followed the instructions in Section 4, Hardware Configuration, you can continue with downloading the application in MyDoors project to a controller.

If you do not have access to a controller or IO modules, you can still follow this example by using a **SoftController**. If you prefer to run with a SoftController; jump directly to the subsection, Setting the System Identity in Control Builder on page 82.

Firmware Upgrade

The controller firmware and Control Builder must be of compatible versions. If you are unsure, perform the steps in this section.

Firmware can only be upgraded when the controller has no application and no previous hardware configuration. If the controller already has an application and hardware configuration; you have to first reset the controller to remove all old applications and configuration information.

Firmware upgrade can be performed from the control builder via the Ethernet network (see the Control Builder Online Help). In case of network failure the serial line upgrade procedure is always available.

Serial Firmware Upgrade Tool cannot be used for firmware upgrade of PM891. The firmware upgrade of PM891 can be done using an SD card or from the Remote System dialog in Control Builder.

Firmware Upgrade via the Serial Cable (TK212A)

1. Connect the serial cable between the Control Builder PC and the controller, as specified in Table 2. For the type of cable, see Appendix D, Communication Cables.

Table 2. Cable connection for the Controller.

Controller	Tool Port	Connector	Cable Name
AC 800M	COM 4	RJ 45	TK212A

⚠ No program capable of blocking the selected COM port, is to be running during upgrade procedure. This applies in particular to the MMS Server program.

2. Turn on the power to the Controller.

3. Select **ABB Start Menu > ABB Industrial IT 800xA > Engineering > Utilities > Serial Firmware Upgrade.** The dialog as in Figure 42 appears.
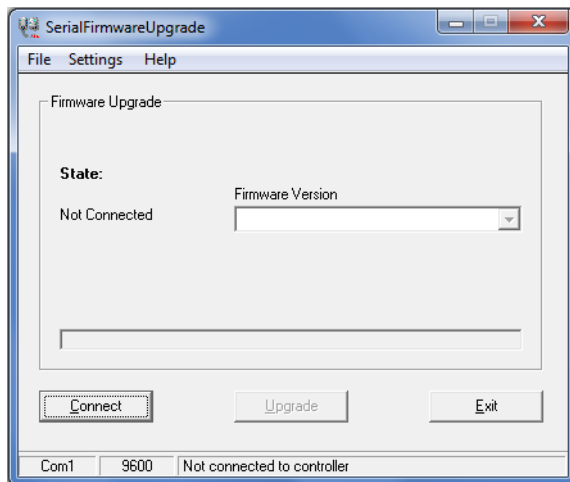
Figure 42. The Serial Firmware Upgrade dialog box.

4. Select **Settings > COM Port** from the drop-down menu. Make sure the settings correspond to the physical COM port, on the PC to which your cable is connected.

5. Click **Connect** and then press the **Init** push-button on the Controller until the **R**un LED starts to blink. Wait about a minute until a message appears. If connection was successful, a confirmation text will occur in the Firmware Version text field (Figure 43).
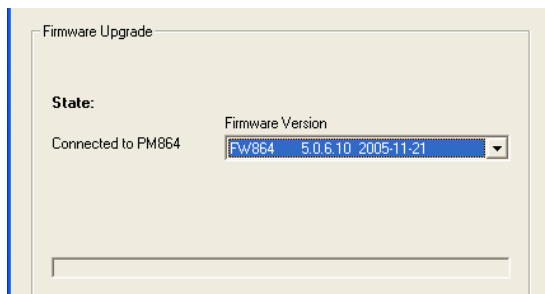

Figure 43. A Firmware version displayed in the text field.

> In the event of an error message "Connection failed", you must check the cables and repeat these steps again.

6. Select Firmware version[1] from the drop-down menu and click **Upgrade**. File transmission starts to the controller. This operation may takes a few minutes. A confirmation window opens when the controller is upgraded, see Figure 44.
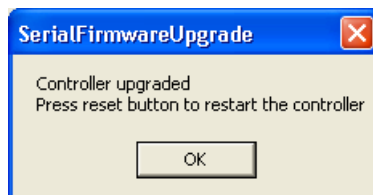
Figure 44. The Serial Firmware Upgrade window.

7. Click **OK**.

8. Click **Exit**.

9. Press the **Init** push-button on the Controller until the **R**un LED starts to blink.

---

1. The firmware version must be supported by the installed Control Builder version.

Setting an IP Address

A unique controller IP address must be set in order to avoid conflict with other devices on the Control Network. This subsection guides you to assign an IP address for your controller via the serial cable (TK212A) without being connected to the network. Furthermore, you will be instructed to setup the PC machine running your Control Builder. However, these instructions are strictly Microsoft Windows specific. A configuring tool, named IPConfig, is used to set the IP address for the Controller.

Setting IP Address for Controller

If you performed Firmware Upgrade on page 75, you can skip the preparations (identical instructions) and go directly to Starting the IPConfig Tool on page 79.

Preparations

Connecting the cable between the Control Builder and the Controller are exactly the same as described in Firmware Upgrade on page 75.

1. Connect a serial cable between the Control Builder PC and the Controller, as specified in Table 3. For the type of cable, see Appendix D, Communication Cables.

Table 3. Cable connection for the AC 800M Controller.

Controller	Tool Port[1]	Connector	Cable Name
AC 800M	COM 4	RJ 45	TK212A

(1) The tool port COM4 is part of controllers PM85x, PM86x and PM891.

2. Turn on the power to Controller.

Starting the IPConfig Tool

3.   Select **ABB Start Menu > ABB Industrial IT 800xA > Engineering > Utilities > IPConfig.** An IP Config dialog opens.
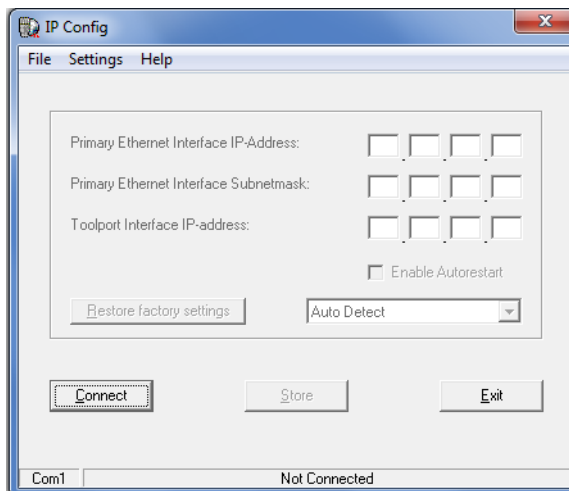

Figure 45. The IP Config dialog box.

4.   Click **Connect** and then press the **Init** push-button on the Controller until the **R**un LED starts to blink. Wait about a minute until the default IP address appears, see Figure 46.
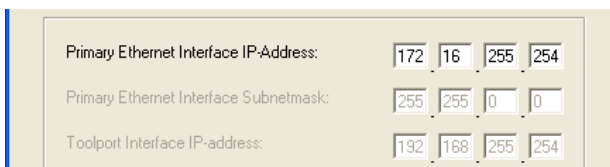

Figure 46. The IPConfig dialog box with factory default setting.

In the event of an error message "Connection failed", you must check the cables and repeat these steps again.

5.   From the IP Config dialog menu, select **Settings > Advanced Mode**.

6.   Enter a unique IP address (obtainable from your Control Network Administrator). Example: IP address 172.16.84.124, see Figure 47.
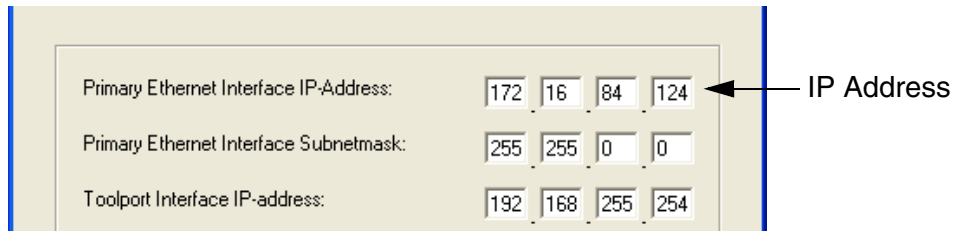
Figure 47. IP Config window for setting unique IP address.

7.   Set Subnetmask 255.255.252.0 and then click **Store**. The new address will be sent to the Controller and an IP Config window re-opens, see Figure 48.
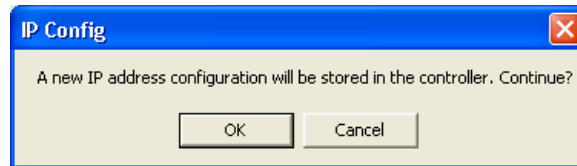

Figure 48. IP Config confirmation dialog window.

8.   Click **OK**.

9.   Click **Exit**.

10. Press the **Init** push-button on the controller until the **R**un LED starts to blink. The new IP address is not valid until the controller has been restarted.

Setting IP Address for PC

To setup the IP address (in Windows) for the Control Builder PC:

1.   Go to **Control Panel** > **Network and Internet** > **Network and Sharing Center** > **View network status and tasks**.

2.   Click **Change Adapter Settings**.

> You must setup the correct order of network adapters in the Control Builder PC; the Control Builder will use the IP address of the last network adaptor. Hence, the Control Network adaptor must come last.

3.   Right-click **Local Area Connection** and select **Properties**. The Local Area Connection Properties dialog opens.

4.   Select **Internet Protocol Version 4 (TCP/IPv4)** and click **Properties**. An 'Internet Protocol Version 4 (TCP/IPv4) Properties' dialog opens.

5.   Select **Use the following IP address.**

The PC and controller NetID must be the same for the first three positions (start from left to right). For example, if the Controller has the IP address 172.16.84.124, then the PC must have the IP address 172.16.84.Q. The number represented by Q must not be same as the Controller; thus **not** 124, 0 nor 255 in this example.

6.   Enter an IP address, in this example (172.16.84.*120*) and then enter sub net mask (255.255.252.0).

The number *120* in this example is arbitrary; you may choose any number in range of (1-254) except number 124 for obvious reasons in this example.

7.   Click **OK**, and close the Local Area Connection Properties dialog.

8.   Connect a network cable. The port and channel positions are shown in Table 4.

To check that the IP configuration works; open the command prompt DOS window and **ping** the Controller. In this example write the following command: *ping 172.16.84.124*.

If the Controller is to be connected to a PC via a switch or hub, then a *straight-through* Ethernet cable should be used. If there is a direct connection between the Controller and the PC, then use a *cross-over* Ethernet cable.

Table 4. Channel positions for connecting the Ethernet cable in the Controller.

Controller	Communication Interface	Position	Channel
AC 800M	Built-in	-	CN1

CN2 port on the Controller must not be connected to the network. This port is used for connecting the Controller to a secondary network.

Downloading the Project via Ethernet

Provided that you have downloaded firmware upgrade (Serial Firmware Upgrade) and given IP addresses, you should have contact with the controller.

This also means that you are ready to download projects to the Controller and Go Online. For downloading to an AC 800M HI controller, see Downloading an Application on page 127

When you use environments, you first change from the Engineering Environment to the Production Environment

Change to the Production Environment

This section is only valid when using environments.

To change to another environment, you re-open the project in the relevant environment.

1.   From the Project Explorer, select **File > Open Project**.

2.   In the displayed dialog box, select the Production Environment.

3.   Click **Open**. The dialog box now displays available projects.

4.   Select the MyDoors project.

5.   Click **OK**. The project is opened in the Production Environment.

Setting the System Identity in Control Builder

In order to download projects to the controller you must first set the system identity in Project Explorer.

Setting the IP address for Controller

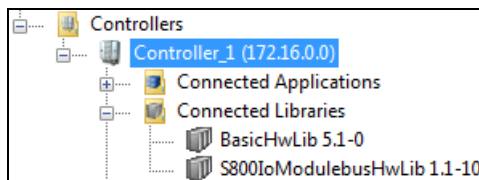1.   In the Project Explorer, expand **Controllers**.

Figure 49. The Controllers expanded in Project Explorer.

2. This time the controller is not reserved. Right-click **Controller_1** and select
    Reserve.

3. The displayed dialog box is used to reserve relevant entities. In this example,
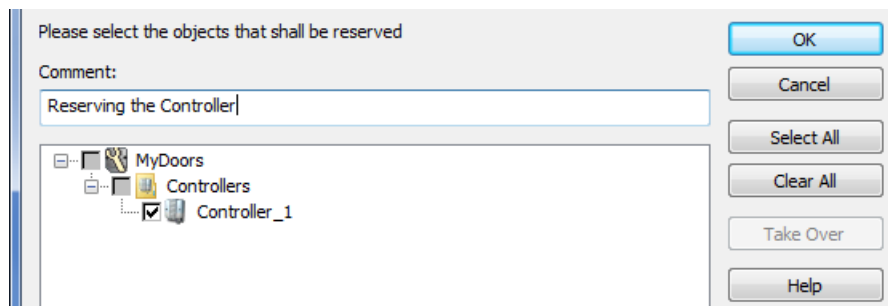   the default selection is sufficient. However, you can enter a comment.

Figure 50. Reserving the controller.

4. Click **OK**. The controller is reserved.

5. Right-click **Controller_1** and select **Properties > System Identity** from the
    context menu. The System Identity window opens.
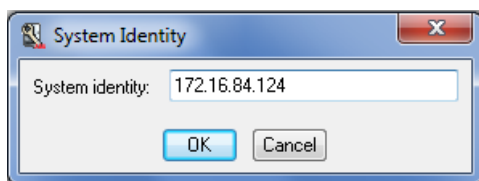
Figure 51. The System Identity window for setting the IP address.

6. Enter the actual IP address of your controller and click **OK**. The System
    Identity window closes.

If you run with a **SoftController** then type in your computer IP address and finish with colon and the digit 2. Example: 10.46.35.117:2

When the computer is started without network connection, the Control Builder and SoftController get the IP addresses 127.0.0.1 and 127.0.0.1:2 respectively (irrespective of fixed IP addresses in the computer). These addresses can be used in the project.
But, without network connection, it is not recommended to use the fixed addresses in the computer, such as 172.16.84.123:2, because when it is used in the project and a downloaded to the SoftController, it results in a crash.

7. Expand **Hardware AC 800M** until you find **1 Ethernet**. Right-click the **Ethernet** icon (at position 1) and select **Editor**. The editor opens.

8. Select the **Settings** tab (lower left corner, see Figure 52) and enter the IP address in the IP address Value field.

Parameter	Value	Type	Unit	Min	Max
IP address	172.16.84.124	string			
IP subnet mask	255.255.252.0	string			
Network Area	0	dint		0	35
Path Number	0	dint		0	3
Node Number	0	dint		0	500
Network Area Local	false	bool			
Send Period	1	dint	s	1	60
Max Lost Messages	3	dint		1	10
Proxy router	0.0.0.0	string			
Target address	0.0.0.0	string			

Settings / Connections / Unit Status /   Row 1, Col 1   NUM

Figure 52. IP address for controller Ethernet port at position 1.

Note that the IP address of the first Ethernet port has to be the same as the IP address of the Controller (system identity). The second Ethernet port (at position 2) is only used if the controller is connected to a redundant network.

9. Click **Save and Close** .

10. You are now ready with all necessary configuration changes, and can release the reservation of the controller.

    Right-click **Controller_1**, and select Release Reservation.

11. In this example, the default selection is sufficient. Click **OK**. The reservation is released.
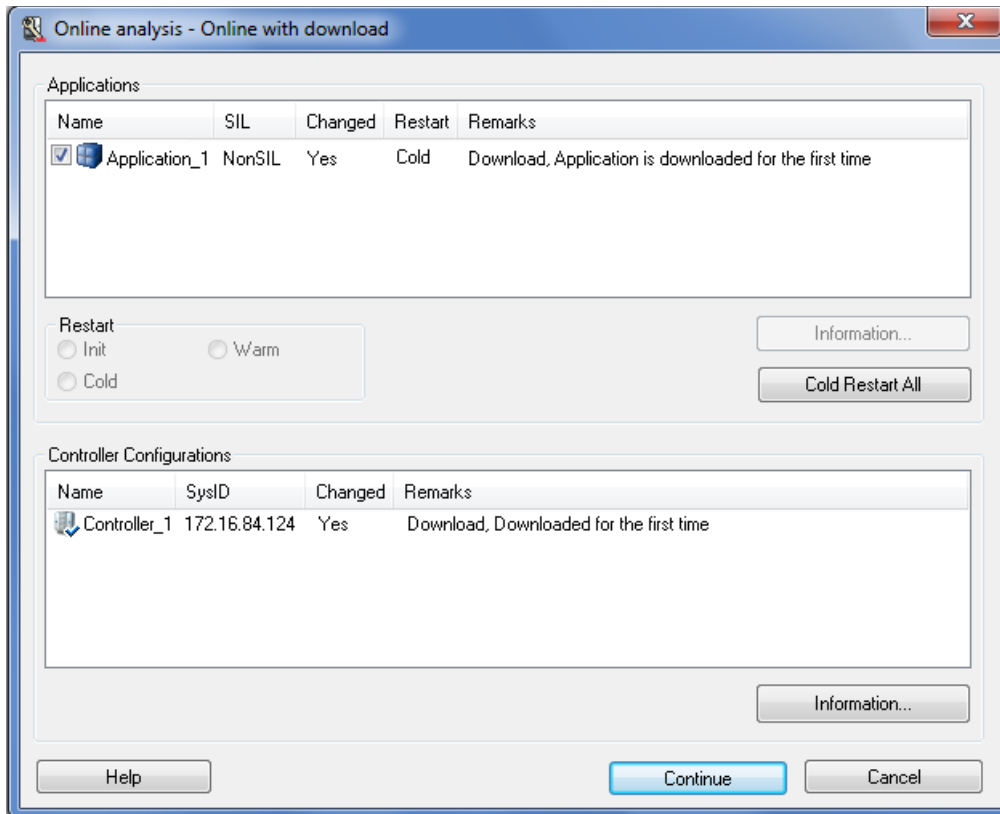
Downloading the Project to the Controller

When you have tested your project and ensured that there are no errors, you are ready to download your application to the Controller.

If you run with a SoftController; go to Downloading to the SoftController on page 87.

Downloading to the Controller

The following instructions address the project MyDoors, which was previously created in Section 3, MyDoors Project. However, these instructions are common for downloading any project application.

1. Make sure MyDoors project is in Offline mode.

2. Click **Download Project and Go Online** . The Online analysis window opens.

3.   Click **Continue**.

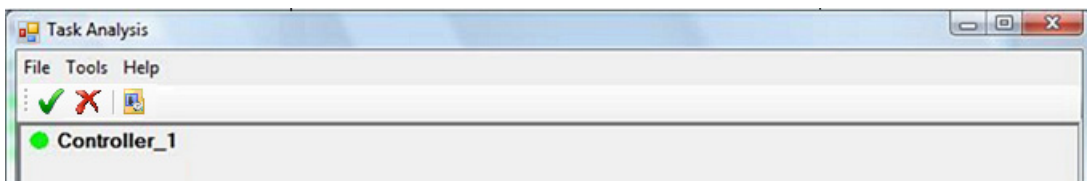4.   In the Task Analysis dialog, click ✔ to accept the download.

Figure 53. Task Analysis dialog

You should now be online.

Downloading to the SoftController

Make sure that your (MyDoors) project is in Offline mode (not running in test mode).

1. Enable the Hardware Simulation feature in Control Builder. See Simulation on page 97.

2. From Project Explorer, expand the Controllers folder.

3. Right-click Controller_1 and select Simulate Hardware from context menu.

Next, you must start the SoftController. Double-click the SoftController icon on the desktop (if desktop shortcut is selected during installation), or from the Start menu on the Windows Task Bar:

The SoftController panel opens.

4. Click the **Start** button. The Status field displays *Started* and the SoftController starts.

From Project Explorer:

5. Click **Download Project and Go Online** . The Online analysis window opens.

6. Click **Cold Restart All**.

7. Click **Continue**. You should now be online.

Test the Diagram Online

This subsection describes how to run the diagram online by forcing the variable Photo cell that is connected to the IO unit DI810 in MyDoors project example.

The *forcing* function can be used to activate/deactivate an I/O.

1. In Project Explorer, right-click **Diagram2** and select **Online Editor** to open the online editor.

2. Right-click I/O module **DI810** and select **Editor**.

3. Click the box in the **Forced** column. Change the variable Photo_Cell value to 1 (true), return quickly to 0 (false) and inspect the motor's values in the online editor (values will change to 1 for five seconds and then return to 0).

Previously, you changed the variable `Photo_Cell` values in the online editor to start and stop the motors. Now, you are using the I/Os to control the motors.

4. Close all editors.

5. Click the ⬛ icon to go to Offline mode.

Forcing I/O Values

In online mode, an I/O channel can be forced to a certain value from Control Builder. The forcing of I/O values is not possible with High Integrity controllers.

• When an input I/O channel is forced, the value that is passed from I/O units to the application is set to the forced value, no matter what the real value from the external device is.

• When an output I/O channel is forced, the value that is passed to the external device is set to the forced value, no matter what the real value from the application is.

This can be used for both testing and fault-finding purposes, as well as to control the process.

What next?

You have almost reached the end of getting you started with Control Builder. You should proceed next with Section 6, View Live Data in Plant Explorer. However, as mentioned before in the beginning of this manual, these sections help you to get started with Control Builder.

The subsequent appendices in this manual also contain useful information that you might need in your daily work with the Control Builder. Control Builder also comes with an extensive set of online manuals presenting in detail the many functions and design issues for building applications. You will be well prepared and experienced with Control Builder and the Project Explorer after referring these manuals.

Furthermore, most of your decision-making that comes via context menus and editors in Project Explorer can be explained and decided from the Control Builder Online Help. To activate the online help, press the F1 key.

Section 6 View Live Data in Plant Explorer

This section is intended to introduce you to the Plant Explorer interface. It is the last and final section with the example MyDoors. You have learned how to test an application locally in Control Builder. Furthermore after configuring an AC 800M controller with S800 IO units you have downloaded the application to the controller and then study how to control the motors online via I/O. Finally this section will help you studying how the variable values from MyDoors project appears as live data from the controller. This is especially interesting since these values will illustrating the same variables that otherwise would be connected to faceplates in an Operators Workplace.

OPC Server

Setting up Variable Communication

Normally, before you create a project you should setup a control network and connect an OPC Server to communicate between the controller and the Operator's Workplace. However, because of the project (MyDoor) that you created previously in Project Explorer, Control Builder has already made a control network for you in Plant Explorer. Thus, the only thing left is configure the OPC Data Source Definition aspect and then connecting the OPC Server to the controller before subscribing live data from the controller.

This subsection assumes that the OPC Server has already been installed and that you have the MyDoors project downloaded and running in a controller. For information on how to configure an OPC Server, see the *System 800xA Getting Started (2PAA111708*)* manual.

If the OPC Server has been installed properly and activated, you should be able to see the OPC Server icon in the Windows Notification Area.
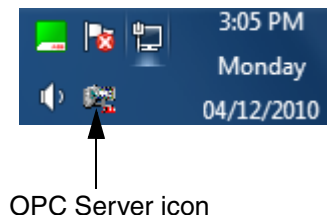
OPC Server icon

Connecting the OPC Server to a Controller

From the Windows Start menu:

1. Select **ABB Start Menu > ABB Industrial IT 800xA > Control and IO > OPC Server for AC 800M > OPC Server for AC 800M**

2. In the **Data Access** tab, enter the controller IP address and click **Connect**.

If your controller is a SoftController, the IP address is the IP address of your PC, with the suffix ":2". For example, if the PC has the IP address 172.16.37.35, the IP address of the SoftController will be 172.16.37.35:2

To retrieve the computer IP address:

a. In the Windows **Search**, type *cmd* and select **Command Prompt** from the list.

b. A DOS window is displayed.

c. Type *ipconfig* and press **Enter**. The IP address is displayed.

d. Write down the IP address and close the DOS window.

3. In the **Alarm and Event** tab, enter the controller IP address and click **Connect**.

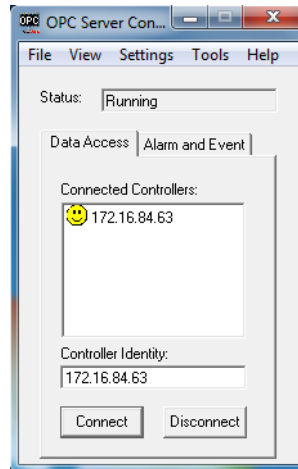The OPC Server should be up and running according to Figure 54.

Figure 54. OPC Server menu.

Plant Explorer

Starting the Plant Explorer

From the **ABB Start Menu** select **ABB Industrial IT 800xA > System > Workplace**. The ABB Workplace Login window opens.
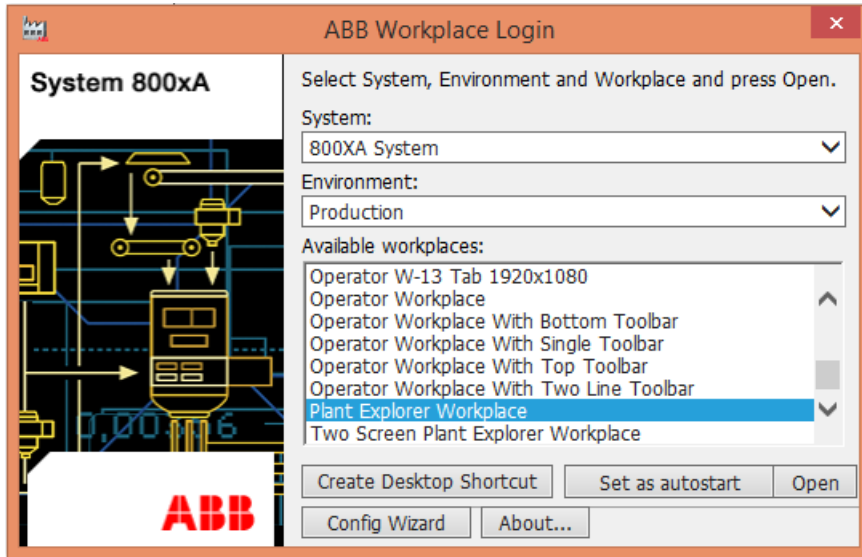
Figure 55. ABB Workplace Login window.

1.   Select **Plant Explorer Workplace** and click **Open**. The Plant Explorer opens.

To access the Plant Explorer Workplace from the desktop, select **Plant Explorer Workplace** and click **Create Desktop Shortcut** (see Figure 55).

2.   Select **Control Structure** from the drop-down menu.
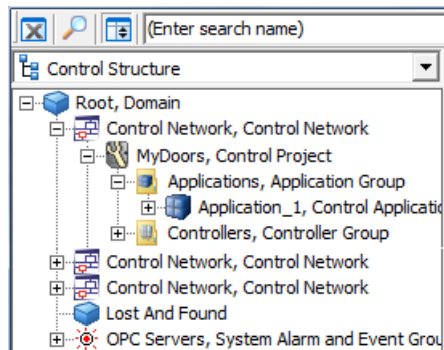
Figure 56. Control Structure view. Note that Control Network and MyDoors project have been added to the Control Structure by Control Builder.

Configure OPC Server in 800xA System

OPC Server for AC 800M must also be configured in the 800xA System:

1. In the Control Structure, select the Control Network icon. An aspect list opens.

2. In the aspect list, click the **OPC Data Source Definition** aspect. The Aspect preview pane opens.

3. Click **New**. A 'New Service Group' window opens.

4. Click **Add**. An 'Add Service Provider' window opens and lists the PC network names.

5. Click **OK**.

Subscribing Controller Data

Make sure you are in the Control Structure, see Figure 56.

1. Expand **MyDoors > Applications > Application_1 > Diagrams** until you see **Diagram2**.

2. Select the control aspect for Diagram2. It is the aspect **Diagram** in the aspect pane. The aspect preview pane will open.
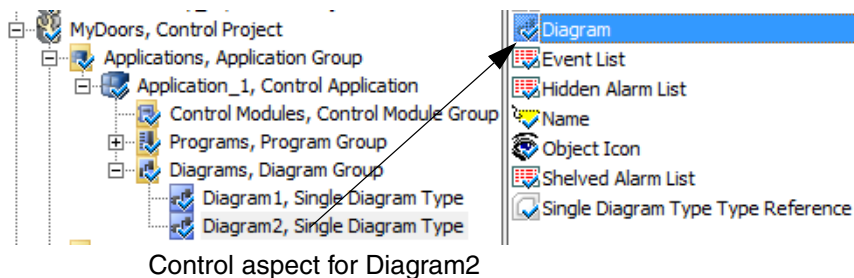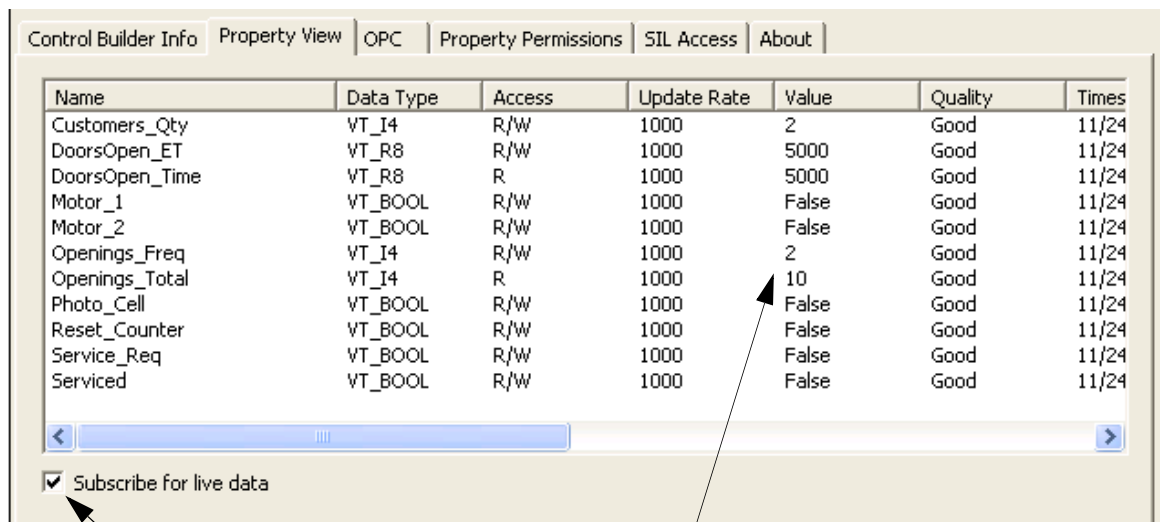

Control aspect for Diagram2

Figure 57. Diagram1 and the control aspect Diagram

3. Select the **Property View** tab in the Aspect preview pane.

4. Check **Subscribe for live data**. Live controller data is now presented in the Aspect preview pane, see Figure 58.

Name	Data Type	Access	Update Rate	Value	Quality	Times
Customers_Qty	VT_I4	R/W	1000	2	Good	11/24
DoorsOpen_ET	VT_R8	R/W	1000	5000	Good	11/24
DoorsOpen_Time	VT_R8	R	1000	5000	Good	11/24
Motor_1	VT_BOOL	R/W	1000	False	Good	11/24
Motor_2	VT_BOOL	R/W	1000	False	Good	11/24
Openings_Freq	VT_I4	R/W	1000	2	Good	11/24
Openings_Total	VT_I4	R	1000	10	Good	11/24
Photo_Cell	VT_BOOL	R/W	1000	False	Good	11/24
Reset_Counter	VT_BOOL	R/W	1000	False	Good	11/24
Service_Req	VT_BOOL	R/W	1000	False	Good	11/24
Serviced	VT_BOOL	R/W	1000	False	Good	11/24

Subscribe for live data is checked

Live controller data

Figure 58. Aspect preview pane with live controller data.

Appendix A Functions and Settings

This section describes some of the functions that are specific to Control Builder. It will also provide guidelines for different settings and configuration issues typical to working with control software for AC 800M. A more detailed description for these topics can be found in the manual *AC 800M Configuration (3BSE035980*)*.

Ready-made Projects for CB Professional

The 800xA system comes with a number of examples that are designed to help you understand how to use various parts of the system and the standard libraries. All these examples are installed as (afw) files in the folder **Program Files (x86)/ABB Industrial IT/Engineer IT / Control Builder M Professional/Examples**.

- An alarm and event example (`AlarmSimple_M.afw`),

- A number of control loop examples (`ControlExamples.afw`),

- A group start example (`GroupStartExample1.afw`),

- A supervision example (`AreaExampleProject.afw`)

- An INSUM example (`INSUMExample.afw`),

- A basic shop door example in FBD, ST, and FD languages (`ShopDoors_FBD.afw`, `ShopDoors_ST.afw`, and `ShopDoors_FD.afw`),

- A tutorial (`Tutorial_03.afw`).

Import an Example to 800xA

Before you can use any of these examples that have been storage on your disk, you must first import them to the Plant Explorer workplace.

If Environments are enabled, the files are imported to the Engineering Environment (default settings).

To import an example:

1. Click the icon (Launch Import Export tool) in the Plant Explorer tool bar.

2. On the **File** menu, select **Open**. The Open Import/Export File dialog is displayed.

3. Browse to the file you want to import, select it and click Open. The Import/Export window opens.

It is also possible to open the Import/Export window by double-clicking the .afw file of the example project from the folder. In this case, ignore Step 1 to Step 3.

4. On the **Actions** menu, select **Import All**. A project, containing the example, is created under a separate Control Network.

5. In the Control Structure, drag the project icon from the Control Network that was created to your "ordinary" Control Network and drop it.

6. Delete the Control Network that was created.

You can now study the example like any other control project.

Testing

Test mode can be used for offline testing of applications. Executing code in Test mode means that the code will be compiled and executed locally in the PC. You can use the online editors to view variables and application execution. The execution time will be much faster than executing code online, and you do not need a controller. However, external communication is disabled during test mode.

Test mode is enabled from the **Tools** menu in Project Explorer (select **Test Mode**).

If the project includes several controllers, you will be asked to choose which controller to start the test mode with. See also Download Project to Selected Controllers on page 105. When Test mode is selected, a version check is performed

and the Test Mode Analysis dialog displays the present applications and controller configurations. Different restart options will be offered in the Analysis dialog, a more defined presentation can be found in Application Restart Mode on page 112.

Simulation

Simulation means that the code is downloaded and executed in a *simulation* controller. Simulation can be done either in a SoftController, or in a *simulation* controller.

> SoftController requires a separate software license.

Simulation can be used to test your applications without connecting them to the physical environment, and to make sure they work and behave as expected.

> When an application is simulated, all I/O copying is cancelled. This means that you need to write corresponding code for I/O responses. For more information see Running in a Simulation Controller on page 100.

You are advised to read the subsection Restrictions Concerning Hardware Simulation on page 101, before you start simulating the hardware.

Select **Tools > Setup > Station > Application Download** to open the Setup - Application Download dialog (see Figure 59).
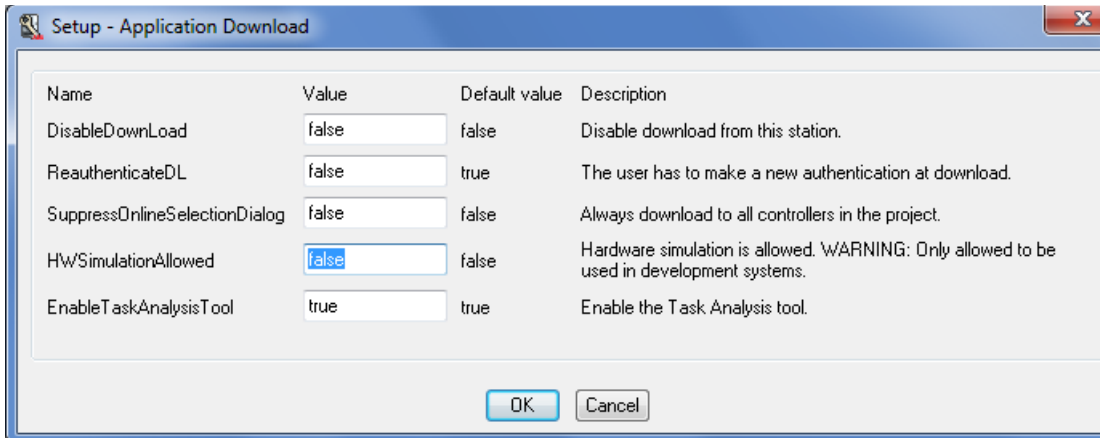
Figure 59. Application Download with default settings

Set the parameter *HWSimulationAllowed* to **true**, to allow hardware simulation.

Simulation Controllers

Applications written for another product type may be tested in a *simulation* controller. The normal copying of I/O-channel data to and from any "real" I/O-module can be replaced by simulated I/O values.

I/O values can be simulated in a number of ways, from simply using logged data, to using advanced process simulations. Since there is no I/O copying in a simulated controller, it is possible to write simulated values directly to I/O variables.

Simulation controllers are always marked with a "Simulated" flag ( ), to eliminate the risk of downloading simulation code to a production controller.

A *simulation* controller accepts a controller configuration even if it does not match its family and type. The following hardware simulation combinations are possible:

Table 5. Matrix of controllers that can be used for simulation

Type of controller to be simulated	Simulation Controllers		
	AC 800M	SoftController	SoftController (HI Mode)
AC 800M	Yes	Yes	No
AC 800M HI	No	No	Yes

Note that an AC 800M High Integrity controller can *never* be used for simulation purposes. An AC 800M High Integrity controller can *only* be simulated in a SoftController High Integrity (SoftController HI).

Applications in Simulation Controllers

Normally, applications that are to be simulated should have a "Simulated" mark.

However, it is always possible to run applications that do not have the "Simulated" mark in a *simulation* controller. The main purpose of the "Simulated" mark on applications is to prevent the application from being downloaded to a controller that is not intended for simulation.

Mark Controller for Simulation

To mark the controller "simulated", right-click the controller and select **Properties > Simulate Hardware** in the context menu.

All hardware units belonging to the *simulation* controller will automatically be simulated-marked, including hardware units such as communication interfaces on the controller's CEX-bus. When a controller is set in simulated mode the icons in the Project Explorer's hardware tree will be exchanged with special simulation icons (with a red "S").

Mark Application for Simulation

To mark the application "simulated", right-click the application in the Project Explorer, and select **Properties > Simulated**.

If the application is simulated-marked, all programs, control modules, function blocks etc. will be simulated, even if they are not simulated-marked. If a simulated-marked program, control module, or function block is used in an application, then the application itself must be simulated-marked as well. It is the simulated-mark on the application that is important.

The simulate-mark on a program may be a way to indicate where, for example, a simulate marked function block is used.

Download to Simulation Controller

To download the hardware configuration and the simulated-marked application to the *simulation* controller, select **Tools > Download Project and Go Online**. See also Difference Report on page 107.

Before download, the controller has to be reset, or already be simulated-marked. To reset the controller, press the controller's INIT button in more than 3 seconds, or temporarily disconnect battery and power supply from the controller.

A simulated-marked application or object can only be downloaded to a *simulation* controller. But, an application without simulation mark can be downloaded to a *simulation* controller.

Running in a Simulation Controller

When an application is running in a *simulation* controller, value and status of I/O variables can be set. The status can be set both on module and channel level.

To test the application in the simulated hardware environment, you can for example use the *forcing* function to change variable values (that will normally be connected real I/O-channel values). Right-click the I/O module and select **Editor**, and in the Status tab, check the box in the Forced column and change the desired variable value. See Figure 60. Note the word "Simulated" in the title bar.
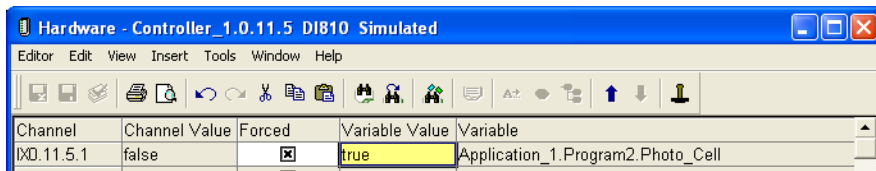

Figure 60. Variable values can be forced to simulate I/O channel values.

The online editor can for example be used to inspect the result of changed variable values for a simulated program.

Restrictions Concerning Hardware Simulation

The hardware simulation functionality is intended for use in a separate testing Control Network. Thus, the hardware simulation controllers shall never be run on the same Control Network as the real production controllers.

If the hardware simulation controllers are located on the same Control Network as the production controllers, either the production controllers may receive unintentional downloads or the future downloads to the production controllers may be affected. There is also a chance of the simulated data being sent to the production controllers, along with the real data. If there is an IP address conflict, the simulated data can also be interpreted as real data by the controllers.

Download

This section describes download and the checks and reports associated with download.

General Download

When you select **Tools > Download Project and Go Online**, the project in Control Builder will be downloaded to the controller and Control Builder will enter Online mode.

Version and Online Analysis

During the version check, the project in the Control Builder is analyzed and compared with the project downloaded in the controller (if any).

The version check will detect if:

*   project versions are identical, and neither the application nor the controller configuration has been changed, then the project will not be downloaded. The effect will be same as going online without download.

- there is no project version mismatch, but the application or controller configuration has been changed, then the changed parts will be downloaded, following the general procedure described in the following.

- the project in the controller contains an application that is not part of the project to be downloaded, but has been downloaded as part of the same project, then this application will be deleted during the download process.

- there is another project in the controller, different from the one to be downloaded, you will have to delete the project and restart the controller, before you can download the new project. See Download New Project to Controller on page 104.

The Online Analysis dialog displays the present applications and controller configurations, and whether or not they have been changed in the Control Builder. An application or controller configuration is considered changed if the version in the Control Builder is different from the version running in the controller.

Compilation

Compilation is performed in Control Builder. If any warnings or errors are detected during the compilation, a Compilation Summary dialog shows a summary of the warnings and errors. You can then choose (if there are no errors) to continue or cancel the compilation.

Compiler switches can be used to set extra restrictions for the code. For more information, see Compiler Switches on page 110.

Change Analysis

Control Builder will perform a change analysis if you have changed:

- variables, function blocks, or control modules,

- data types, function block types or control module types,

- libraries,

- applications.

The change analysis is performed, before downloading, to check the possibility of maintaining variable values after restart.

The change analysis detects mismatches between the application version in the controller and the application version to be downloaded.

A mismatch can occur if:

- A variable has been assigned another data type,

- A variable, function block or control module has been renamed,

- A data type, function block type or control module type is missing, has been renamed, or has been moved to another library,

- A library has been given a new name (this will result in a mismatch for all data types, function blocks types and control module types from this library),

- An application has been renamed (this will result in a mismatch for all data types and variables, function blocks and control modules in the application).

For variables with attributes *Retain* or *ColdRetain*, the change analysis is performed in the following way:

1. All data types, function block types, and control module types, which existed before the change, are checked for name matching.

2. All variables, function blocks, and control modules are checked for name and type matching.

If the change analysis detects mismatches, Control Builder cannot determine how to retain variable values. A warning dialog will display information about detected mismatches. You may then have to guide Control Builder and correct mismatching names, by giving the renamed object the new name (click **Rename** in the dialog).

Download and Go Online

The changed parts of the project (application and/or controller configuration) are downloaded to the controller(s). The controller(s) will stop the running application(s), and restart with the new/changed versions, and with variable values maintained (depending on the type of attribute and restart).

After download is completed, Control Builder enters Online mode. In Online mode, Control Builder communicates with the controller(s), and you can view variables and application execution in the controller(s) using online editors. Furthermore, you can issue operations to the controller.

> ℹ️ If the message "Download aborted. See the controller log for further information." appears during download, an error has been detected in the downloaded controller configuration. A common cause is that there is not enough controller memory. You may find details in the controller log. If the controller is still running, you can try to compile and download again. See the manual *AC 800M Configuration* for how to locate the log file.

Download New Project to Controller

When you select **Tools > Download Project and Go Online**, the Confirm Deletion of Project dialog is displayed if there is another project in the controller. To reset and restart the controller, click **Complete Reset** in the dialog. See Figure 61.

> 💡 Another way to reset and restart the controller is to press controller's INIT button for more than 3 seconds.
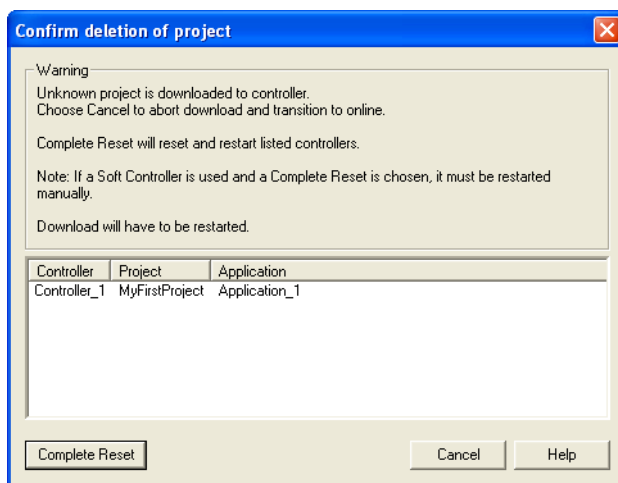
Figure 61. Confirm Deletion of Project dialog

> ℹ️ When downloading to a High Integrity controller, a complete reset cannot be done from this dialog. The **Complete Reset** button will be dimmed.

The controller will then be reset, all existing applications in the controller will be deleted, and the controller will be restarted. The download of the new project can then be continued (as described in General Download on page 101.)

Download Project to Selected Controllers

For a project that contains more than one controller, it is possible to choose which controllers to download and go online with. In this way, certain applications and controller configurations can be excluded from the download, and you can go online with only a sub-set of the project to a selected controller. This reduces the compilation time.

When working in a multi-user environment, one user can work with some parts of the project, while other users are working with other parts. If the other parts of the project cannot be compiled since they are not finished, the user will still be able to go online with parts that are finished.

> If an application is connected to several controllers, it is not possible to select only one controller. The remaining controllers will be added automatically.

If you select **Tools > Download Project and Go Online** for a project with more than one controller, the Selection of Controllers dialog will be displayed. For example, in Figure 62, "Controller_2" and "Controller_3" cannot be separately selected or excluded since one application is connected to both these controllers.
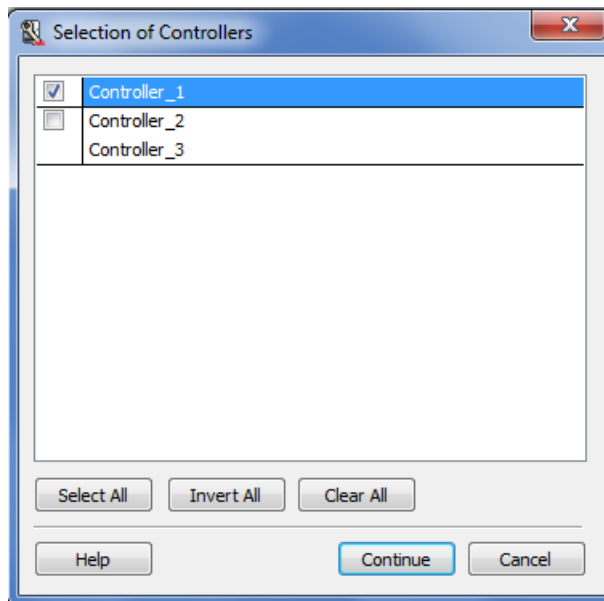
Figure 62. Selective of Controllers dialog

Another way to select a specific controller is to right-click the controller in Project Explorer, and select **Download and Online Mode**.

The Selective Download function is by default enabled, but can be disabled by selecting **Tools > Setup > Station > Application Download**, and in the Setup dialog setting the parameter *SuppressOnlineSelectionDialog* to true.

Selecting an Application to Download

Besides connecting several applications to one controller and download them, you can also select just one of these applications for download. This is convenient when some parts, for example in Application_1 are not ready but everything in Application_2 is finished and ready for testing. If both applications are connected to Controller_1, you will be giving a chance to select if both applications should be downloaded or just Application_2.

1.   Click **Continue**-button in Figure 62, an Online Analysis window will open.

2.   Select the application to be downloaded. See the example in Figure 63.
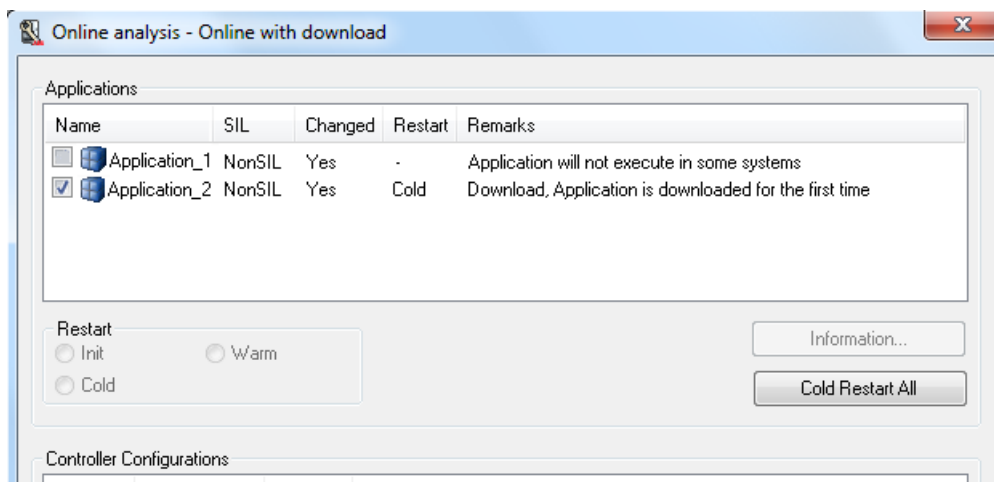
Figure 63. Connected applications to a controller. Only Application 2 will be downloaded.

Difference Report

Difference report shows the difference between data downloaded to the controller and the data present in Control Builder. The tree view shows the parts of the application that have changed. By clicking an item in the tree, you can display the present controller code, and the new code beside. Differences are also indicated by colors (the color coding is explained on the status bar at the bottom of the report window).

If the Difference Report function is enabled, it will display:

• Difference report,

• Source code report.

Based on the information presented in the reports you can either accept or reject the changes that is if you want the download to be carried out or cancelled.

The function is enabled/disabled from the root (Project icon) at the top of the Project Explorer tree structure.

Right-click the Project Icon (e.g. MyDoors in Figure 13) and select **Settings > Difference Report** from the context menu. A 'Difference Report Settings' window will open.
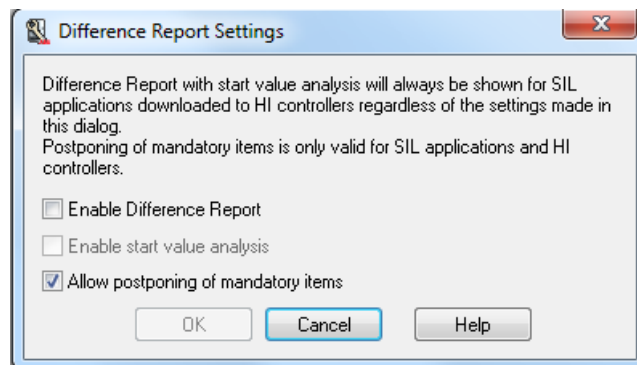
Figure 64. Difference Report Settings window

To enable the difference report, select the *Enable Difference Report* check box and click **OK**.

The Difference Report function is always enabled when downloading any application to a High Integrity controller.

Difference Report Before Download

If the difference report setting is enabled, the Difference Report Before Download dialog (as shown in Figure 65) appears before the download of the application to the controller. Click ✔ to accept the download.
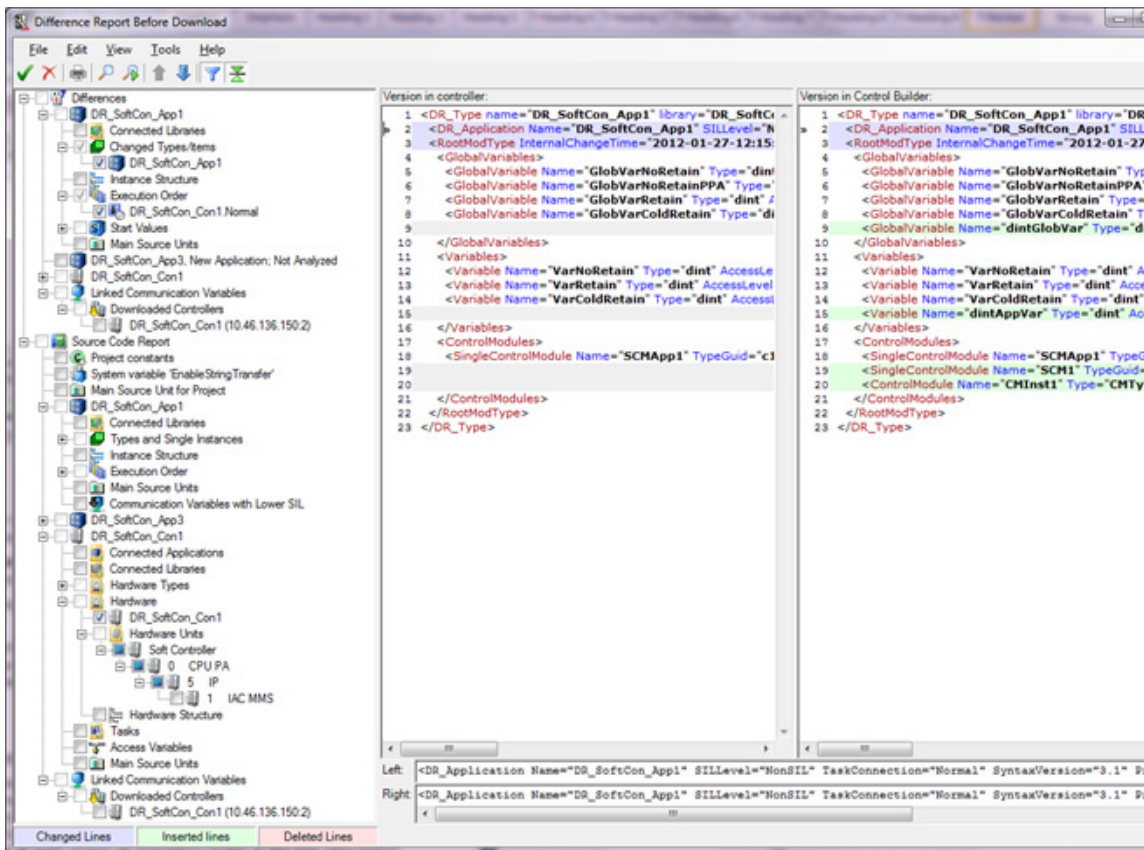
Figure 65. Example of a Difference Report Before Download

Re-Authentication

The Re-Authenticate function is used to ensure that a user is authorized to download a project. If the function is enabled, a dialog is shown and you will be asked to enter name and password before the download is carried out.
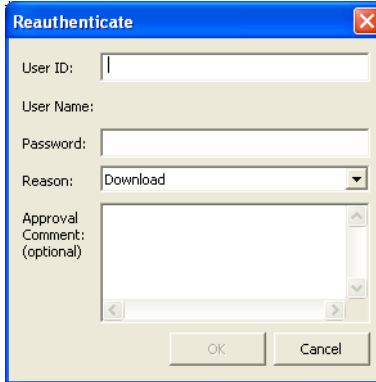
Figure 66. Re-authenticate dialog

Disable/Enable Re-Authenticate function

1. Select **Tools > Setup > Station > Application Download**.
    A 'Setup Application Download' dialog opens.

2. Set the parameter **ReauthenticateDL** to *true/false*. The function is enabled per Control Builder station.

Compiler Switches

Configurable switches in the compiler allow you to introduce extra restrictions on code. The restrictions are divided into:

- Global restrictions
- SIL restrictions

Global restrictions are valid for all code. SIL restrictions are valid only for code marked SIL1-2 and SIL3. For SIL1-2 and SIL3 code, there are also mandatory restrictions, which cannot be changed.

A restriction can have the values:

- Allowed
- Warning
- Error

Mandatory SIL1-2 and SIL3 restrictions are always errors.

The restrictions are checked during the check of each type, and also during the compilation. If you change a restriction, the project will be re-compiled when you perform a download of the project, and the code will be validated with the new restriction.

Restrictions makes it possible to warn against or forbid things that may result in complex code and thereby errors. For example, to check the usage of Instruction List language, you can change the corresponding switch from "Allowed" to "Warning", and you will get a compilation warning if there is any IL code in the project. Another example is when a new library is imported, you can then turn all switches from "Warning" to "Error" to check the quality of the new library.

Compiler switches are configured in the Compiler Switches dialog, which is displayed if you; right-click the Project Icon (e.g. MyDoors in Figure 13) and select **Settings > Compiler Switches** from the context menu. A 'Compiler switches' window will open.
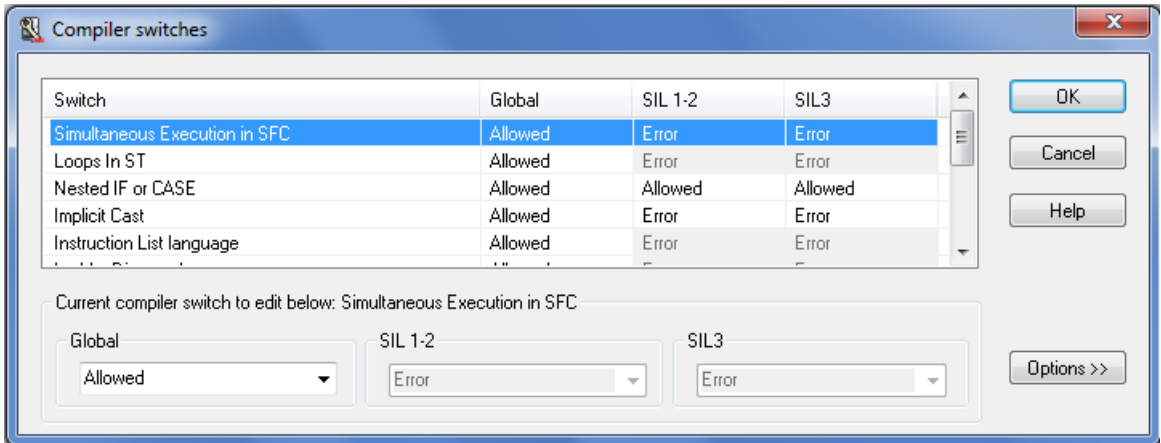
Figure 67. Compiler switches dialog

It is possible to exclude a library from a certain restriction, provided that the restriction is a warning. If a library is excluded, no checks are performed on the restriction for any type in that library. For example, if a new external library results in many warnings, you may want to filter out acceptable warnings to make it easier to read other warnings. Click **Options** in Compiler Switches dialog, and in the expanded dialog select the library to exclude from the selected restriction.
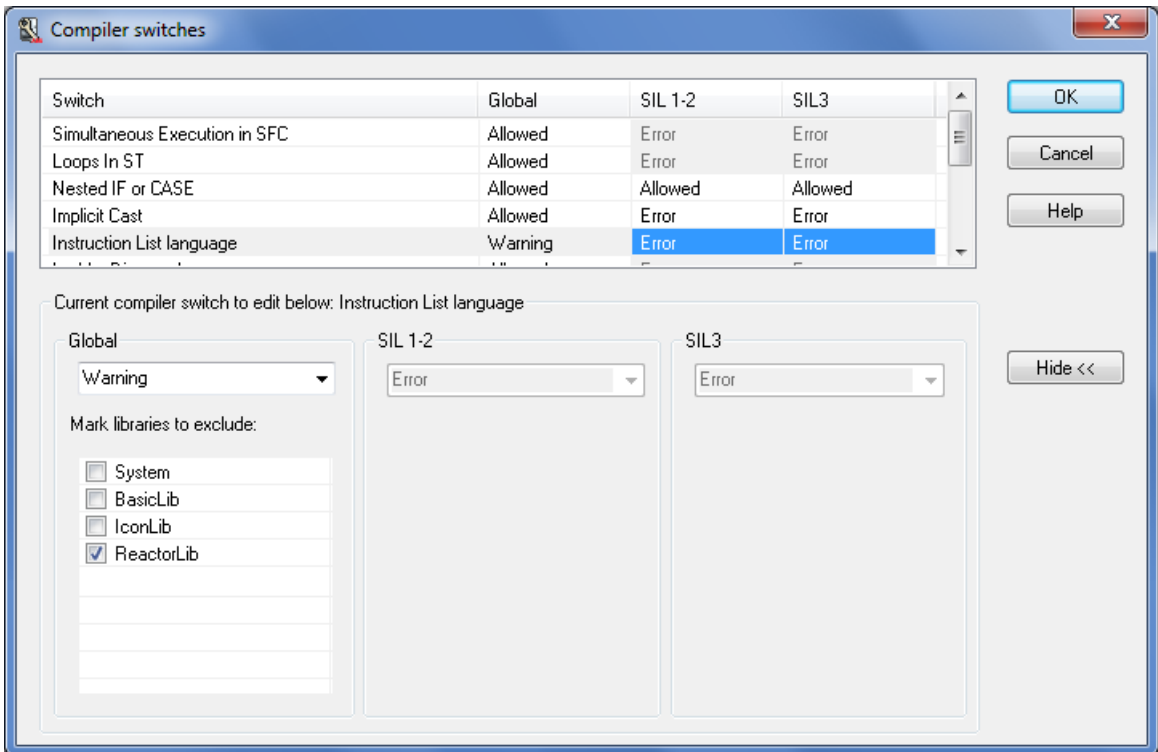
Figure 68. Compiler switches, excluding library from restriction.

Application Restart Mode

Variables can be given the attributes *retain* or *cold retain*. Depending on the attributes specified for the variables, the values of the attributes are either maintained or lost upon warm restart, cold restart, and power failure. Control Builder sets the attribute *retain* on all variables by default.

It is also possible to specify an *initial value* for a variable. The initial value is used if the value of the variable is not maintained after a restart or power failure.

For more information about variables, attributes and values, see the manual *AC 800M Configuration*.

Warm Restart

A warm restart is performed after a download of application changes and after a short power failure (provided that it is backed up by the battery in the controller).

At warm restart, only variables with the attribute *retain* or *coldretain* are retained. The values are retained from the current version of the application running in the controller to the new version, based on name matching.

All other variables are set to their initial value, if such values have been given. Otherwise the values of the variables are set to the default values of the data types. The default value is "false" for boolean, 0 (zero) for integer, and so on.

It is possible to force a cold restart after a download, even if a warm restart is proposed.

After a power fail, instead of doing a warm restart of the application SIL3 applications are restarted using cold retain marked values saved in the controller periodically with a cycle time set by the user.

Cold Restart

A cold restart is always performed after the first download. It is also possible to order a forced cold restart after a download (in the Online Analysis dialog, select the application and select **Cold Restart**, or click **Cold Restart All** to perform a cold restart of all applications).

A short (less than 3 seconds) push on the controller's INIT button also results in a cold restart.

At cold restart, only variables with the attribute *cold retain* are retained. The values from the aspect server are only used when downloading from Control Builder using cold restart.

All other variables are set to their initial value, if such values have been given. Otherwise the values of the variables are set to the default values of the data types.

Cold Retain Values

Control Builder normally saves cold retain values to the aspect server when going from Online to Offline mode. The saving is enabled/disabled, via **Tools > Setup > Station > System Variables**, and the parameter *SaveColdRetainOnOfflineTransfer*.

In Online mode, you can order Control Builder to save cold retain values by selecting **Tools > Save "ColdRetain" Values**.

The OPC server can also be configured to save cold retain values to the aspect server, either manually (the values are saved when you click **Save**) or automatically (the values are saved periodically with a cycle time set by you).

When you need to setup the saving of cold retain values, consult OPC Server online help and the manual *OPC Server for AC 800M*, *Configuration*.

Variable values in Download mode

Depending on which mode Control Builder enters Online mode, a cold or warm restart will affect variable values differently. See Table 6, Table 7, and Table 8.

Initial Restart

Table 6 shows how a initial restart affects variable values when Download mode is started.

Table 6. Initial restart, starting values

Variable attribute	Starting value
ColdRetain	Initial value
Retain	Initial value
Other	Initial value

Cold Restart

Table 7 shows how a cold restart affects variable values when Download mode is started.

Table 7. Cold restart, starting values

Variable attribute	Starting value
ColdRetain	Cold retain value
Retain	Initial value (if any)
Other	Initial value (if any)

Warm Restart

Table 8 shows how a cold restart affects variable values when Download mode is started.

Table 8. Warm restart, starting values

Variable attribute	Starting value
ColdRetain	Retain value
Retain	Retain value
Other	Initial value (if any)

Variable values in Test mode

Initial Restart

The variable values are similar as in Download mode, see Table 6.

Cold Restart

The variable values are similar as in Download mode, see Table 7.

Warm Restart

The Warm restart mode can only be selected if the Control Builder previously was in Test mode. The variable values are similar to Warm restart variable values in Download mode if the control builder was not in the Test mode previously, see Table 8.

Warm Restart of the application is performed after the power failure. The restart values of the variables are as shown in Table 9.

Table 9. Warm restart: starting values

Variable attribute	Starting value
ColdRetain	Last test session
Retain	Last test session
Other	Initial value (if any)

Compilation is performed in Control Builder. You can choose (if there are no errors) to continue or cancel the compilation. If compilation warnings or errors are detected, the Compilation Summary dialog will display a summary of the warnings or errors.

Power Failure

A power failure, in combination with low battery capacity in the controller, means that the controller will be completely reset, that is, empty. Or by pressing the controller's INIT button for more than 3 seconds it will have the same effect. You will then have to download the project again.

If you remove the power source (battery) during a power fail, all error logs will be lost.

In these cases, only variables with the attribute *cold retain* can be retained. The values are then retained from the aspect server. All other variables are set to their initial values, if such values has been given. Otherwise the values of the variables are set to the default values of the data types.

Appendix B License Management

Introduction

License Management comprises of the following three licenses:

- Control Builder license

    - A license that controls the number of currently active Control Builder clients, see Control Builder Licenses on page 118.

- SoftControl license

    - Each started Soft Controller instance should have one license each. Several Soft Controllers can be started on the same PC.

- Library License

    - For using the libraries – TCPHwLib, UDPHwLib, and BurnerLib – separate licenses are required, see Library Licenses on page 120.

Information regarding licensing errors and what happens when licenses need to be extended (the system will start showing messages that licenses are missing, so called system wide annoyance mode) etc. can be found in the manual *AC 800M Configuration* (3BSE035980*) (in the Maintenance and Trouble-Shooting section).

Control Builder Licenses

A Control Builder license is included when purchasing an engineering workplace.

When you open or create a new control project, Control Builder checks for a 'free' Control Builder license. If the server cannot find one, a dialog will tell you that the action was denied.

You cannot open or create a new control project without a Control Builder license.

Controller Capacity Points (CCP)

When applications are downloaded to controllers, the total CCP count is calculated.

The total CCP count is calculated by checking the CPU type of the controllers that participated in the download. Other installed hardware units (like IO modules, drives, INSUM units, and so on) are not checked for calculating the CCP.

The following restrictions apply to CCP count calculation for controllers (PM8xx):

• Controllers with the *Simulate Hardware* state is excluded from CCP calculation.

• Redundant PM8xx is counted as single PM8xx.

CCP Calculation Rules

Each CPU type has a unique Controller Capacity Points (CCP) figure that relates to the capacity for the controller. Table 10 provides the CCP for each type of controller.

Table 10. CCP for each controller type

Controller Type	CCP
PM851	13
PM856	25
PM858	30
PM860	40
PM861	50
PM862	58
PM864	67
PM865	67
PM866	100
PM867	100
PM891	150
SoftController	0

See

In a Control Builder project, if an application is downloaded to a controller, the CCP count for this controller is always included in all subsequent downloads. This happens even though this controller is not selected in the subsequent downloads. To remove the CCP count for this particular controller (if this controller is no longer used), remove it from the Control Builder project and proceed with the download to the other controllers.

Library Licenses

The libraries – TCPHwLib, UDPHwLib, and BurnerLib – require separate licenses. These licenses are checked after the download of the project to the controller, in any of the following cases:

- The TCPHwLib is added to the Connected Libraries folder under required controller in Controllers folder.

- The UDPHwLib is added to the Connected Libraries folder under required controller in Controllers folder.

- The BurnerLib is added to the Connected Libraries folder under the required application in Applications folder.

See also Checking the CCP and Library License for a Controller on page 120.

Checking the CCP and Library License for a Controller

To check the required CCP Count and Library License for a controller that participated in the download:

1. In Plant Explorer, go to Control Structure, go to the project and then select the controller. The Controller aspect will be displayed in the aspect pane.
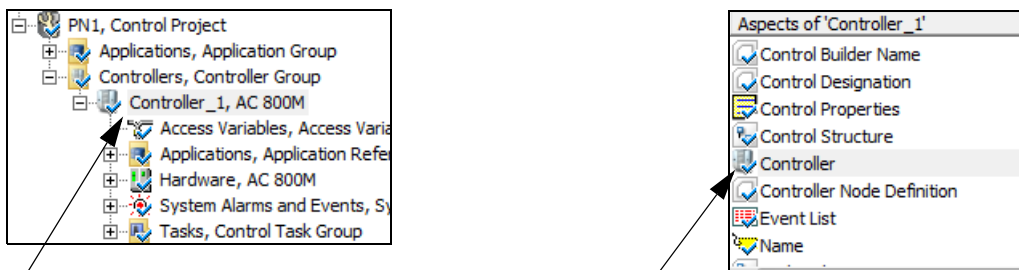
Controller in the Control Structure   Controller aspect

Figure 69. (Left) Selected controller in the Control Structure. (Right) The Controller aspect for the selected controller (Controller_1).

2. Click the Controller aspect and select the Licensing tab in the aspect preview pane. The current downloaded CCP value for the selected controller, and the

Library Licenses are displayed (see Figure 70).
The *Required points* is validated against the actual licenses available.

If the controller has not been downloaded or if it is in *Simulate Hardware* state, the tab displays information about this condition, and no CCP values or Library Licenses are displayed.
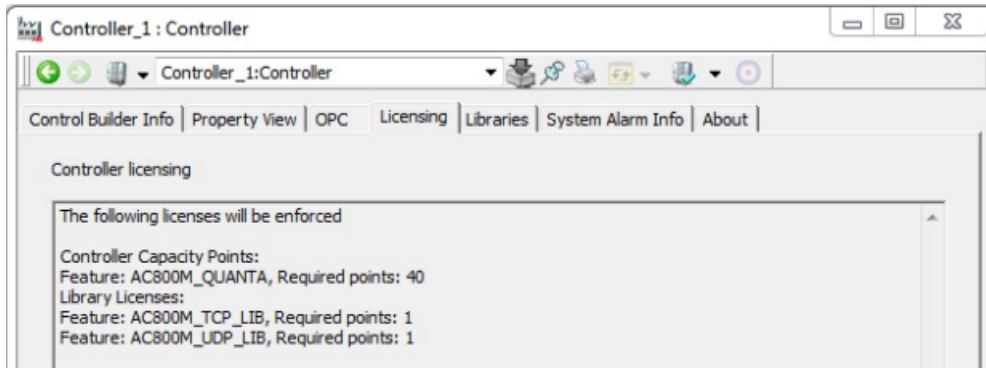
Figure 70. Example of the Licensing tab in the Controller aspect of the selected controller

The Licensing tab in Figure 70 shows the required Controller Capacity Points (CCP) count and Library Licenses for only the selected controller to which the application is downloaded. To calculate the total CCP count, open the Controller aspect of each controller that participated in the download, and add the CCP counts.

Combined AC 800M High Integrity Controller

If an AC 800M High Integrity controller is to be used to run both non-SIL and SIL applications, it is considered to be a combined PA/Safety controller.

A combined PA/Safety controller requires an additional license. For further additional information, please contact your local ABB vendor.

Appendix C SIL Certified Applications

Introduction

SIL stands for "Safety Integrity Level", as specified in the standard IEC-61508. To run SIL certified AC 800M applications, you need a SIL certified AC 800M High Integrity controller.

It is also possible to run non-SIL applications in a High Integrity controller.

Certified applications comply with SIL1-2 and SIL3.

The *Safety Manual*, *AC 800M High Integrity (3BNP004865*)* contains guidelines and safety considerations related to all safety life-cycle phases of an AC 800M High Integrity controller.

SIL Information Can Be Disregarded by Non-SIL Users

Part of the information given in this manual applies to SIL applications only. If so, this will be indicated. If you do not run any SIL applications and do not have an AC 800M High Integrity controller, please disregard any SIL-specific information.

SIL is a standard for applications with very high demands on reliability. SIL applications should only be used where specifically required.

For information on restrictions regarding SIL applications and High Integrity controllers, see online help and the manual *AC 800M Configuration*.

SIL Applications

A SIL application is always marked with a SIL icon when shown in the programming interface.
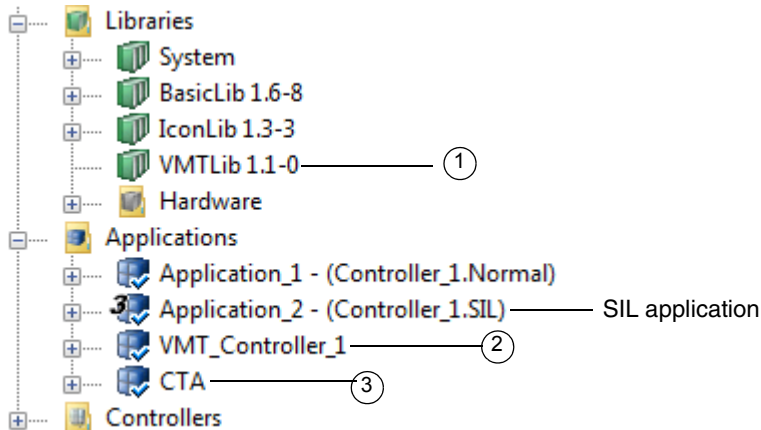
Figure 71. SIL application in Project Explorer.

When an AC 800M High Integrity controller is used, there is always an additional standard library, as well as two additional applications (see Figure 71):

- VMTLib (1) is the Virtual Machine Test library, containing types used when running the VMT application.

- VMT_Controller_1 (2) is an application used to make sure that the HI controller works properly. The name of this application is made up of the prefix VMT_ and the name of the controller, which is called HI_23 in our example.

- CTA (3) is the Compiler Test Application, used to make sure that the compiler works properly.

⚠ Never delete the above libraries and applications if you run a High Integrity controller. The VMT library, the VMT application, and the CTA application are needed to make sure that the High Integrity controller and compiler work properly.

A SIL application may only use library types that are SIL certified. SIL certified library types are marked with SIL icons in the programming interface. The compiler will also check for constructs in your code (for example, loops) that are not allowed

according to the SIL standard. If a SIL-marked application contains not allowed types, objects or constructs, it cannot be downloaded due to compilation errors.

Setting SIL-Levels

You can set Non-SIL, SIL1-2, and SIL3 for the Safety Level on Function Blocks, Control Modules, SingleControlModules, Programs, Diagrams, Applications, and Tasks according to the requirement.

Right-click on the object (except for tasks where the setting is performed in the Task Dialog), select **Properties**, and then select **Safety Level** as shown in **Figure 72.** Select the required SIL Level.
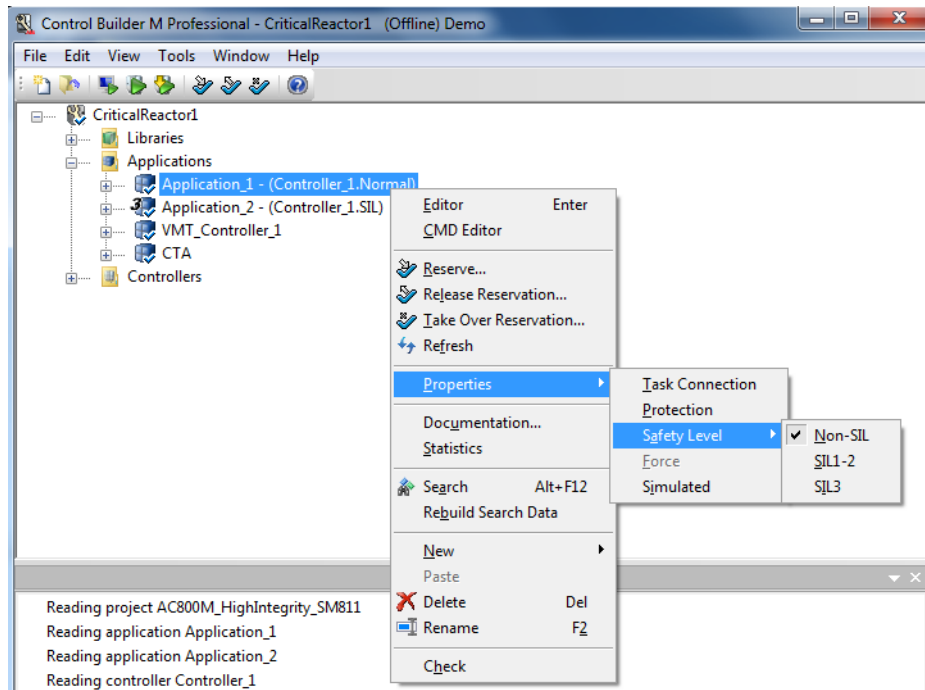
Figure 72. Setting SIL-Level

The SIL-level is displayed as a black number on top of the icon for the intended object. If no number is displayed the level is Non-SIL. This marking is also available on Functions.

Restricted SIL-Level

It is not allowed to use output parameters from Function Blocks or Control Modules marked with Non-SIL in the parameter description in a way that can influence the safety function of a SIL classified application. If such code affects an output from a SIL3 application, it might result in a Safety Shutdown.

Some of the objects and functions in standard libraries have a restricted SIL-marking for both SIL2 and SIL3 functions. These functions are allowed to be used in a SIL application with few restrictions. Using a restricted object or function to affect I/O or to be used in peer-to-peer communication might result in a malfunctioning application or a safety shutdown.

Restricted objects and functions are identified by a special icon in the project tree in the Control Builder project explorer. They must not be used as a part of the critical loop for a safety function. The color of the SIL-digit in the icon is grey on a restricted object and function, compared to the black digit on regular SIL-classified objects and functions. Examples of Restricted icons is as shown in the figure below.

It is not allowed to use Functions, Function Blocks or Control Modules marked as SILxRestricted in a way that can influence the safety function of a SIL classified application.

High Integrity Controllers

An AC 800M High Integrity (HI) controller consists of a PM865 or PM867 processor unit and a number of HI-specific objects that are added to the hardware

tree, see Figure 73.
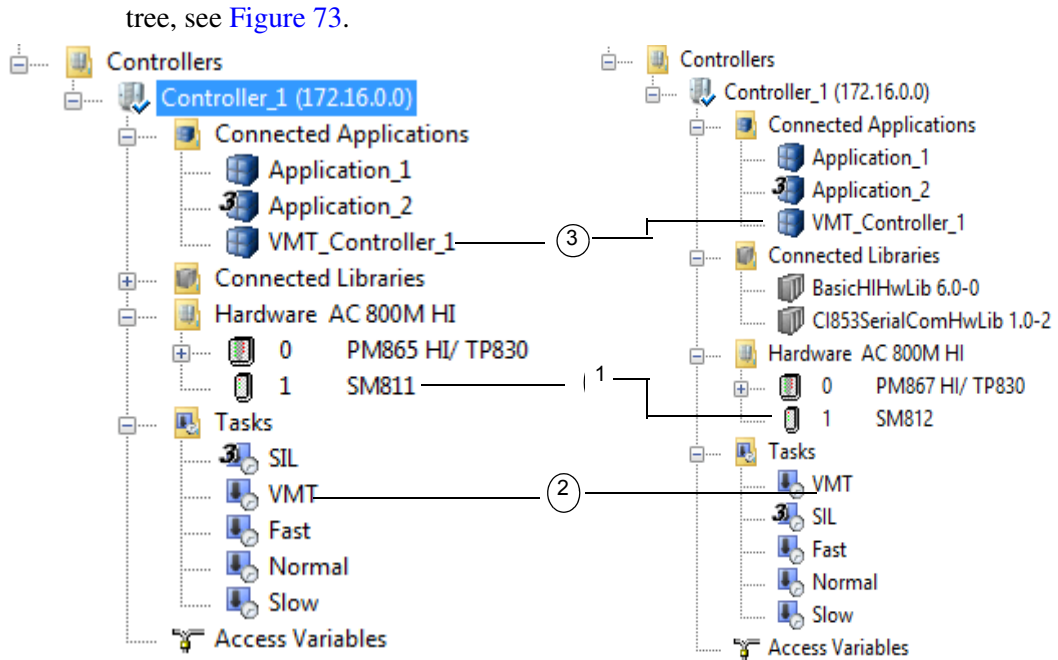
Figure 73. AC 800M High Integrity controller.

The following objects are only added for a HI controller:

- SM81X (1) is a module that supervises the function of the controller.

- The VMT task (2) is used to run the VMT application. Note that the VMT application is connected to HI controllers automatically, and should not be removed.

- The VMT application (3) (see previous subsection) is automatically connected to the controller.

In addition to the objects listed above, a High Integrity controller also has HI-specific firmware, which makes sure that the controller works as specified in the SIL1-2 and SIL3 standard.

Downloading an Application

Access Enable must be set to (ON) before downloading to an AC 800M HI controller, no matter if the applications are non-SIL or SIL marked.

Appendix D Communication Cables

Serial communication between Control Builder and the AC 800M controller is done by using the TK212A cable.

Connect the DB9 Female connector to a Control Builder PC COM port, thus the RJ45 (8P8C) plug to the AC 800M controller COM4 port. The Figure 74 illustrates the TK212A pin-out configuration.

Connecting Control Builder PC to an AC 800M Controller
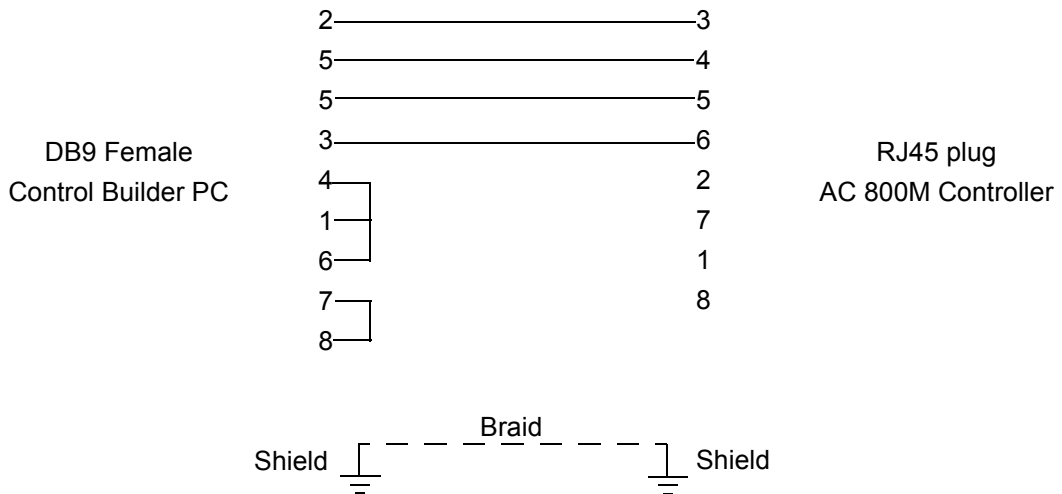
Figure 74. Cable TK212A Pinout configuration.

INDEX

3BSE041880-600 A

Contact us

www.abb.com/800xA
www.abb.com/controlsystems

3BSE041880-600 A

Power and productivity
for a better world™

ABB