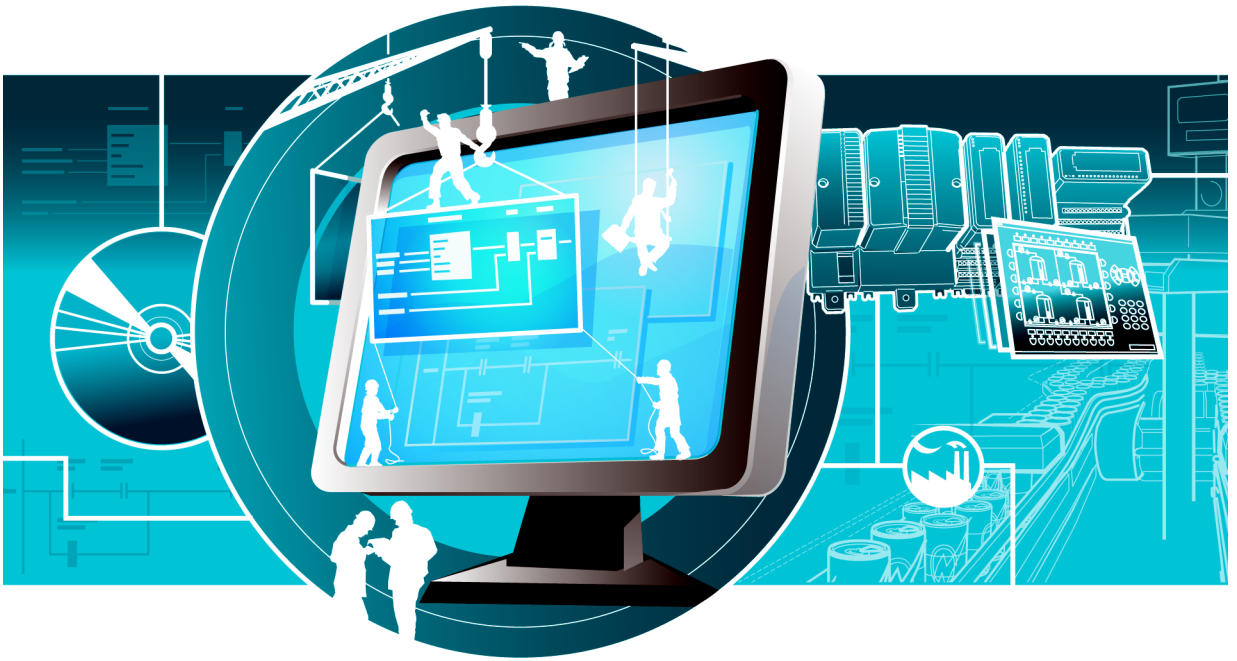


# Industrial<sup>IT</sup> Compact Control Builder Version 4.1

## Getting Started Introduction and Installation





**Industrial<sup>IT</sup>**  
**Compact Control Builder**  
Version 4.1

**Getting Started**  
Introduction and Installation

---

## NOTICE

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.

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 89/336/EEC and in Low Voltage Directive 72/23/EEC.

Copyright © 2005 by ABB.  
All rights reserved.

Release: May 2005  
Document number: 3BSE039838R201

## TRADEMARKS

All rights to trademarks reside with their respective owners.

---

# TABLE OF CONTENTS

## About This Book

Document Conventions .....	10
Use of Warning, Caution, Information, and Tip Icons .....	11
Applicable Specifications .....	11

## Section 1 - Introduction

Documentation Strategy .....	13
Product Overview .....	14
Requirements.....	14
Project Explorer .....	15
Libraries .....	16
Supported PLC and Configurations.....	17
PLC Control Builder Functions.....	18
Not supported Functions .....	19
Multi-User Engineering.....	20
Using Online Help in PLC Control Builder .....	21
Project Documentation.....	21
Online Manuals .....	21
Getting Help in PLC Control Builder .....	24

## Section 2 - Installing PLC Software

Before You Begin .....	28
Step-by-Step Instructions .....	29
Installing the PLC Software .....	29

Starting Up .....	31
Control Builder .....	31
SoftController .....	32
OPC Server .....	33
Configuration Issues.....	34

### **Section 3 - PLC Control Builder User Interface**

Introduction .....	35
About Programs and Projects .....	35
Project Templates .....	36
Project Explorer .....	37
Title Bar, Menu Bar and Tool Bar.....	37
Project Explorer Pane .....	38
Libraries Folder.....	40
Applications Folder.....	40
Controllers Folder .....	41
Message Pane.....	43
Editors .....	44

### **Section 4 - MyDoors Project**

MyDoors Project .....	46
Specifications .....	46
Defined Variables.....	47
Creating MyDoors Project .....	48
Local Variables .....	50
Function Blocks .....	53
Code Blocks .....	55
Code Input.....	57
Testing MyDoors Project .....	64
Project Examples in Control Builder .....	69

---

## Section 5 - Hardware Configuration

Configure Hardware .....	73
Connecting IO to an Application.....	76
Connect Variables to I/O Channels.....	77
Method 1 - Using Dot Notation .....	77
Method 2 - Using a Path Selector.....	78
Reading I/O addresses from the Application .....	80

## Section 6 - Connecting the PLC and Go Online

Firmware Upgrade.....	81
Setting an IP Address .....	84
Setting IP Address for PLC.....	84
Setting IP Address for PC .....	87
Downloading the Project via Ethernet.....	89
Setting the System Identity in Control Builder .....	89
Downloading the Project to the PLC.....	91
Test the Program Online .....	94

## Appendix A - PLC Control Builder AC 800M Settings

System Settings for PLC Control Builder .....	98
Auto Logon to Windows .....	98
Product Settings for PLC Control Builder.....	99
Auto Startup of Control Builder.....	99
Memory Reservation .....	100
Language .....	102
File Locations.....	103
Multi-User Configuration .....	104
Creating a Shared Project Folder .....	105
Setting Up PLC Control Builder Stations .....	107
Setting Up OPC Server .....	109
Configuration Example .....	114
Guidelines for Multi-User Engineering .....	115
Disable/Enable Difference Report.....	116

## **Appendix B - Network Redundancy**

Setting Up Redundant Network .....	117
Two Separate Redundant Networks .....	117
Decide IP Addresses .....	120
Setup Using the IPConfig Tool .....	122
Configure Controller Ports from Project Explorer.....	122
Configure PC Ports in Windows XP Professional .....	123
Download Project and Go Online .....	123
Design .....	124
IP Address .....	124
Separating Client/Server and Control Network .....	127
Summary of Configuration Steps.....	128

## **Appendix C - Upgrade**

Introduction .....	129
Licenses .....	130
Products .....	131
Applications .....	132
Saving Application and Configuration Data .....	132
Remove Products .....	134
Install Products .....	134
Restore Application and Configuration Data.....	135
Downloading the Project.....	139

## **Appendix D - Communication Cables**

Connecting Control Builder PC to a PLC .....	147
--	-----

## **Appendix E - Programming Languages**

General .....	149
---------------	-----

## **Appendix F - Glossary**

## **INDEX**



---

# About This Book

Welcome to PLC Control Builder AC 800M - an effective but still easy to use programming tool. This manual is produced for anyone intending to use the programming tool PLC 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. Consequently you are advised to look up the appendices if you need to learn more about for example upgrading projects, multi-user engineering or network redundancy.

If this is your first time working with a 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 a brief introduction to the document strategy and product overview.

[Section 2, Installing PLC Software](#), helps you install a single user configuration that is, a PLC Control Builder and an OPC Server installed on the same PC machine.

[Section 3, PLC Control Builder User Interface](#), explains the PLC Control Builder and its core interface Project Explorer.

[Section 4, MyDoors Project](#), encourages you to build a small project example, and getting yourself acquainted with the Control Builder environment.

[Section 5, Hardware Configuration](#), teaches you how to add or remove hardware units from the tree structure in the Project Explorer.

[Section 6, Connecting the PLC and Go Online](#), starts with the prerequisites for connecting a PLC and then guide you through downloading a project and Go online.

## Document Conventions

The following conventions are used for the presentation of material:

- The words in names of screen elements (for example, the title in the title bar of a window, the label for a field of a dialog box) are initially capitalized.
- Capital letters are used for the name of a keyboard key if it is labeled on the keyboard. For example, press the ENTER key.
- Lowercase letters are used for the name of a keyboard key that is not labeled on the keyboard. For example, the **space bar**, **comma key**, and so on.
- Press CTRL+C indicates that you must hold down the CTRL key while pressing the C key (to copy a selected object in this case).
- Press **ESC E C** indicates that you press and release each key in sequence (to copy a selected object in this case).
- The names of push and toggle buttons are boldfaced. For example, click **OK**.
- The names of menus and menu items are boldfaced. For example, the **File** menu.
  - The following convention is used for menu operations: MenuName > MenuItem > CascadedMenuItem. For example: select **File > New > Type**.
  - The **Start** menu name always refers to the **Start** menu on the Windows Task Bar.
- System prompts/messages are shown in the Courier font, and user responses/input are in the boldfaced Courier font. For example, if you enter a value out of range, the following message is displayed:

Entered value is not valid. The value must be 0 to 30.

Variables are shown using letters in Italic style.

*MaxLimit*

## Use of Warning, Caution, Information, and Tip Icons

This publication 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 which could result in *electrical shock*.



Warning icon indicates the presence of a hazard which 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 which 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, comply fully with all **Warning** and **Caution** notices.

## Applicable Specifications

This product meets the requirements specified in EMC Directive 89/336/EEC and in Low Voltage Directive 72/23/EEC.



---

# Section 1 Introduction



Due to a last-minute name change, please refer to Compact Control Builder wherever PLC Control Builder or PLC Software is mentioned in this manual or when using the product.

PLC Control Builder is a programming tool for creating PLC based control solutions when using the AC 800M hardware as your PLC hardware.

The PLC Control Builder comes with type solutions for simple logic control, device control, loop control, alarm handling etc. packaged as standard libraries. You can also insert self-defined types from other projects into your current project. The Control Builder supports five different programming languages, Function Block Diagram, Structured Text, Instruction List, Ladder Diagram and Sequential Function Chart according to IEC 61131-3. In addition to this it supports the Control Module language. Other useful functionality is online debugger and test mode.

## Documentation Strategy

Two introducing manuals are available: *Getting Started* and *IEC 61131 Control Languages*.

*Getting Started* (this manual) gives an introduction for new users of the PLC Control Builder. The manual also gives a description of the installation procedure.

*IEC 61131 Control Languages* is a good place to start for learning the basics of the different aspects of automatic control and the control languages following the IEC-61131 standard.

For more information about online manuals and online help in PLC Control Builder, see [Getting Help in PLC Control Builder](#) on page 24.

## Product Overview

PLC Control Builder is a fully integrated Windows XP Professional and Windows 2000 Professional application. It provides tools for programming applications and configure PLC hardware units from the AC 800M hardware family.

## Requirements

The minimum requirements are besides the operating system Windows 2000 Professional or Windows XP Professional; Microsoft Word 2000 or 2002 and Acrobat Reader version 4.0 or later.

Microsoft Word is required for creating user-defined project documentation and Acrobat Reader is required to read online manuals.



For more information, see also [Using Online Help in PLC Control Builder](#) on page 21.

## Project Explorer

The PLC Control Builder core 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 PLC 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, functions, etc.) and the hardware (the actual hardware connected to the PLC) are modelled in a project. The relationships are visualized in Figure 1.

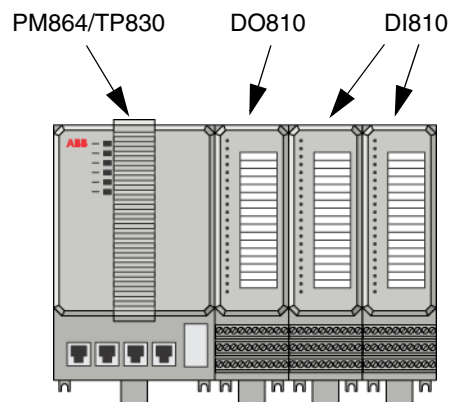
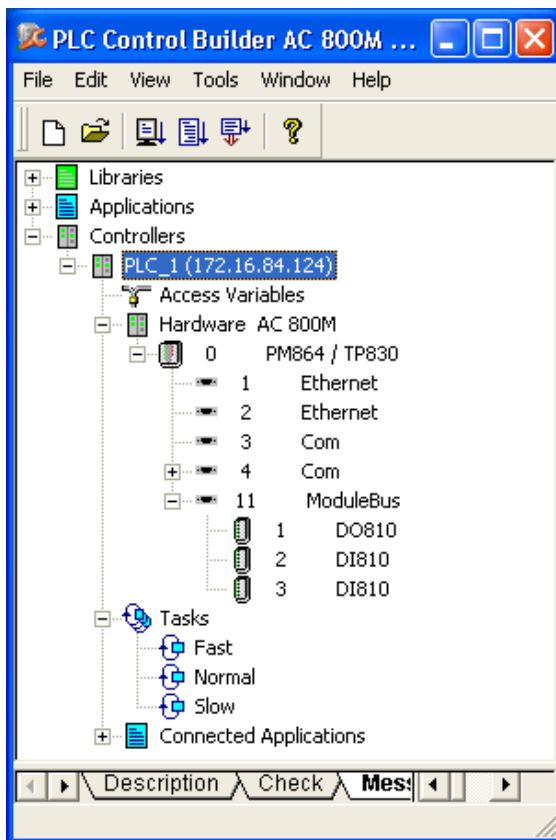


Figure 1. Project Explorer and actual hardware setup.

## Libraries

PLC Control Builder is delivered with an extensive set of predefined type solutions stored in standard libraries. These include data types, functions, function blocks and Control Modules that can be used in your projects.

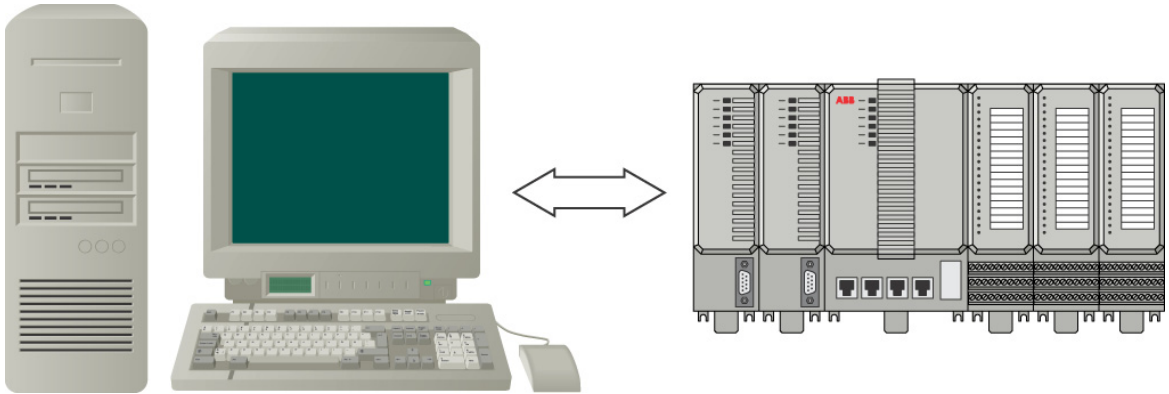
All standard libraries are included during the PLC Control Builder installation and are available in your projects. PLC Control Builder provides the following libraries:

- 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, Foundation Fieldbus, SattBus, COMLI and Siemens 3964R protocols.
- The Control Libraries, they include single PID control and cascade PID control function blocks, control modules, etc.
- The Alarm and Event Library, it contains function blocks for alarm and event detection, and alarm printouts on a local printer.



## Supported PLC and Configurations

The PLC is the “target” for your applications which means that you download your application to the PLC from the Project Explorer. The programming code is then executed in the PLC.



*Figure 2. The PLC Control Builder station communicates with a PLC.*

## PLC Control Builder Functions

The PLC 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 PLC Control Builder can handle up to 256 applications where each application can handle 64 programs at the most. A maximum of 32 Control Builder PCs can be used together in multi-user environment and up to 32 PLCs 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, which are components for object-oriented (and graphical) programming.

[Table 1](#) lists the most common PLC Control Builder functions at glance.

*Table 1. Main PLC Control Builder Functions*

<b>Functions</b>
Backup/Restore
Create/change/insert libraries
Create/change/use Control Modules
Difference report (between previous/new application)
Distribute code in an application to several controllers
Downloading projects and go online
Multi-user engineering
Search and Navigation Tool
Testing projects off line

## Not supported Functions

When you require additional functionality for building DCS type of control solutions you can use the control builder available in the ABB 800xA DCS system offering. The 800xA control builder (the CBM Professional) adds the following functions to the set of functions available in PLC Control Builder:

- Batch handling.
- Audit Trail.
- SFC Viewer.
- High Integrity Controller for SIL applications.
- CI860 for FF HSE, and CI862 for TRIO I/O.
- Information routing via HART protocol.
- Security (see [Appendix F, Glossary](#)).

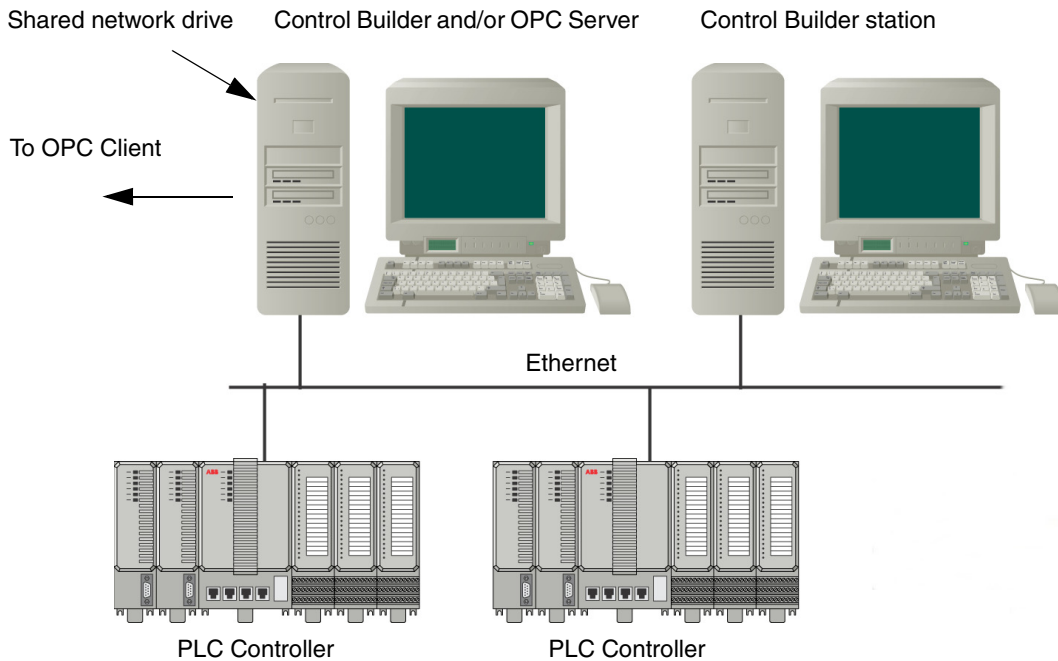


The functionality above is not included in the PLC Control Builder AC 800M.

## Multi-User Engineering

PLC Control Builder supports multi-user engineering with a maximum of 32 separate engineering workplaces. In a multi-user configuration all Control Builder PCs and the OPC Server must have access to the common project file(s). This means that a common Project folder must be created on a shared network server.

The network server can be placed anywhere suitable in your network; thus located and handled by a Control Builder PC, or in an OPC Server PC, or located as a stand-alone file server.



*Figure 3. Programmers can share the same project. Multi-user engineering stores projects on a shared network drive.*

## Using Online Help in PLC Control Builder

The PLC Control Builder provides online documentation.

### Project Documentation

PLC Control Builder facilitates a project documentation feature for libraries, applications, and controllers or for single types. The project documentation will be produced as a Microsoft Word file and can be connected to either a standard document template or user/company specific. A table of contents is automatically generated for every project document.

#### Creating Project Documentation

From the Project Explorer:

1. Right-click an object in the Project Explorer tree and select **Documentation** in the context menu. A Documentation window will open.
2. Click **More** button. An Editor Properties window opens.
3. Select a tab in the dialog window.
4. Click **OK**.

### Online Manuals

The PLC Control Builder provides you with related manuals online.



You need Acrobat Reader to open and read the online manuals provided by the PLC Control Builder product.

#### Accessing Online Manuals

From the Project Explorer:

1. Select **Help > Manuals**. A list of manuals will open.

Table 2. PLC Control Builder Manuals

<b>Manuals</b>	<b>Description</b>
Getting Started, Introduction and Installation	This manual
IEC 61131 Control Languages	Introduction to different aspects of automatic control and the control languages following the IEC-61131 standard
Communication, Protocols and Design	Describes criteria for selecting networks and communication protocols and is intended for engineers who are planning the design of a new network or the expansion of an existing one.
OPC server for AC 800M, Installation and Configuration	Makes acquaintance with the OPC Server for AC 800M, its technical characteristics and standards, as well as how to connect and what interfaces to use.
AC 800M, Hardware and Operation	This book describes in detail the hardware platform AC 800M, together with the controllers and associated units that are used in the AC 800M Controller.
PROFIBUS-DP, Wiring and Installation	This book provides application notes and advice for wiring and installation of PROFIBUS networks.
PROFIBUS-DP, Engineering and Configuration	This book describes the configuration of the PROFIBUS DP-V1 in the 800xA control system using the communication interface CI854/CI854A.
S900 I/O	(a number of manuals provided)
S800 I/O	(a number of manuals provided)

*Table 2. PLC Control Builder Manuals*

<b>Manuals</b>	<b>Description</b>
S200 I/O	This manual is intended for those involved in the configuration, installation and maintenance of the S200 I/O system.
S200L I/O	This book describes how to configure, install and maintain S200L I/O units.

## Getting Help in PLC Control Builder

In PLC Control Builder you can get Help in the following ways:

- Context-Sensitive Help (F1)
- Contents Topic
- Index
- Keyword Search

### Getting context-sensitive Help

You can get context-sensitive Help for items in the Project Explorer. To access context-sensitive Help:

1. Select the element you want help on (any item from the tree, command inside an editor etc.).
2. Press the F1 key.

### Accessing contents topic

Each Pop-up window with a Help button offers an entry into a category of information in the Help content tree.

### 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. 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 topic 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 finds all matches. This means that you have to be rather specific, when using this function, or you will end up with far too many hits.



---

## Section 2 Installing PLC Software

This section explains how to install and start-up a single-user configuration, which means basically a PLC Control Builder and an OPC Server installed together on the same PC station. The software delivered on the CD is divided in two parts - the PLC Control Builder AC 800M and the OPC Server for AC 800M. Each of these is installed with the help of installation wizards.

- The first wizard contains PLC Control Builder, PLC firmware, Base software for SoftController, RNRP and User Documentation.
- The second wizard contains OPC Server for AC 800M.

Note, that you must first run the *PLC Control Builder* installation before running the *OPC Server* installation. Simply follow the default installation instructions given in the Wizard.



The PLC Control Builder can only open projects stored in a Project folder created during the Setup Wizard installation. If you have changed the project folder path in a more recent installation, your previous projects cannot be found by the Control Builder. This problem is easily solved by either changing the project folder path back to previous location or, copy/paste the previous projects, from the Windows explorer, into the current Project folder location.

## Before You Begin

Skim through the following check-list before installing your PLC Software:

- A PC with either Windows XP Professional or Windows 2000 Professional installed. For PC requirements, see [Requirements](#) on page 14.
- Administrator privilege for the login to Windows.
- Remove previous Control Builder versions from the PC<sup>1</sup>. This also includes other products within the Control<sup>IT</sup> suite.

---

1. From Windows Control Panel select **Start > Control Panel**. Select **Add or Remove Programs** from the list.

## Step-by-Step Instructions

You must install PLC Software from the CD onto the local disk, you cannot run any of the software from the CD.

### Installing the PLC Software

1. Log in as Administrator in Windows.
2. Insert the CD into the drive. After a few seconds the Welcome dialog window will appear (Figure 4). If the dialog box does not appear, start the file *Startme.bat*, located in the root directory of the CD.



Figure 4. The Welcome PLC Software dialog window.

The installation dialog window contains the following buttons:

- The **Release Notes** button gives you the latest information.
- The **Install Software** button activates the installation procedure.
- The **Installation help** button accesses information on how to install a product.
- The **Exit** button allows you to quit the installation procedure.

### Installing the PLC Control Builder

1. Click the **Install software** button (see Figure 4). A software installation window opens.



You should always start with the PLC Control Builder installation first because an OPC Server installation needs to read the Control Builder settings that was created previously during the PLC Control Builder installation.

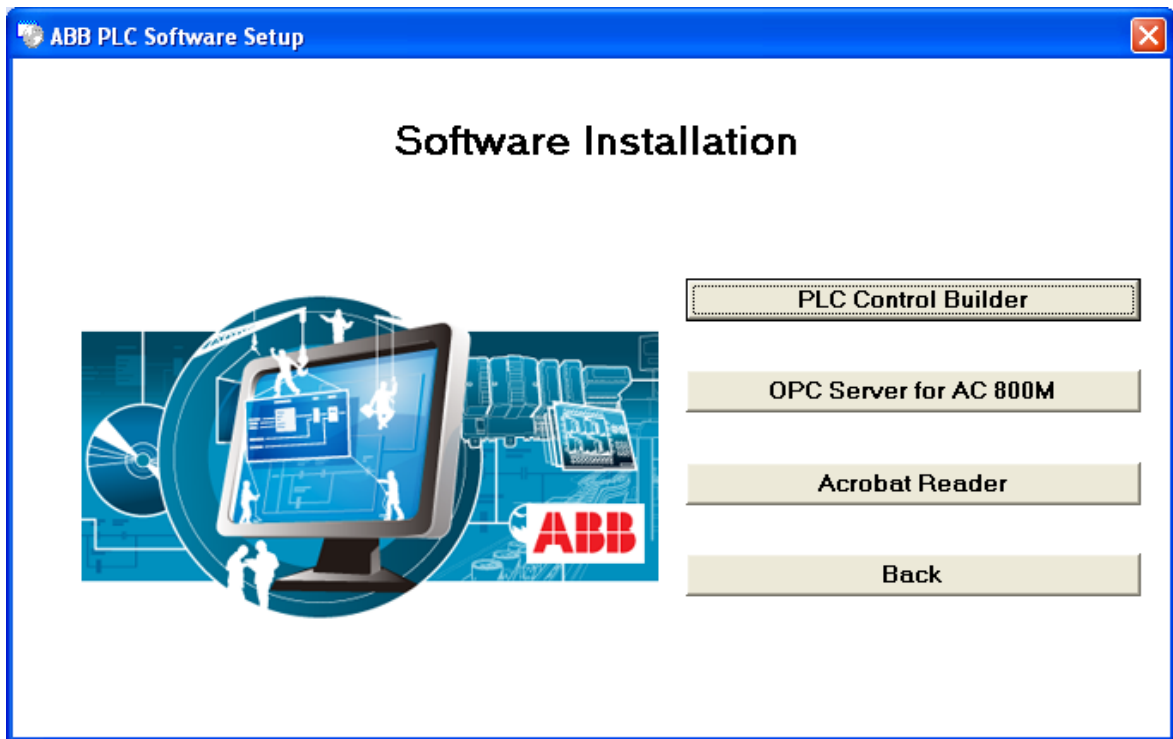


Figure 5. The Software Installation dialog window.

2. Click the **PLC Control Builder** button. The Installation Wizard starts.
3. Follow the on-screen installation instructions.



Using the Cancel button in any of the installation Wizard dialogs will interrupt the installation. When installation procedure is interrupted, all previously installed components will be disregarded.

### Installing the OPC Server for AC 800M

1. Click the **OPC Server for AC 800M** button (see [Figure 5](#)). The Installation Wizard starts.
2. Follow the on-screen installation instructions.

Running the OPC Server on the same PC as the PLC Control Builder does not require further settings.



For more information about setting up an OPC Server for multi-user engineering, see [Setting Up OPC Server](#) on page 109.

## Starting Up

### Control Builder

#### Starting the PLC 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, **Start > All Programs > ABB Industrial IT PLC > AC 800M > PLC Control Builder AC 800M**

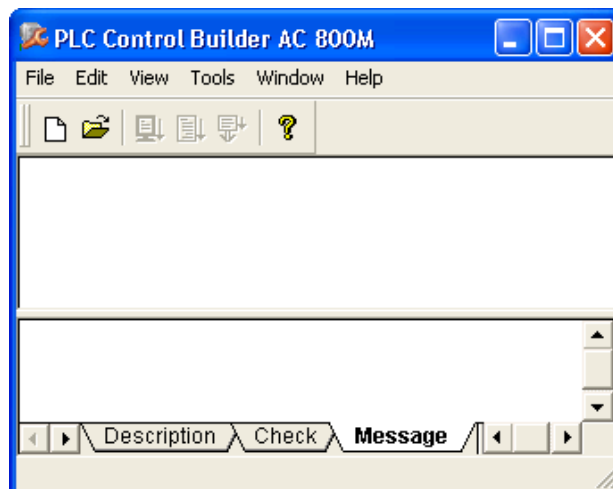


Figure 6. PLC Control Builder started.

## SoftController

The SoftController is a simulation tool that runs with Base Software for SoftControl (in short). A SoftController let you download projects from the Project Explorer even though you may not have access to a real PLC. Instead of downloading to a PLC, you can download to the SoftController.



In order to start the SoftController, you must either have administrator privileges in Windows, or be part of the local group *ABB Controller user group*. Contact your administrator and apply to be a member of this group. The *ABB Controller user group* is created automatically in Windows during the SoftController installation.

The following steps help you start the SoftController and to locate its network address (the address must be set in Project Explorer and OPC Server panel).

### Starting 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,

**Start > All Programs > ABB Industrial IT PLC > AC 800M > SoftController > SoftController**

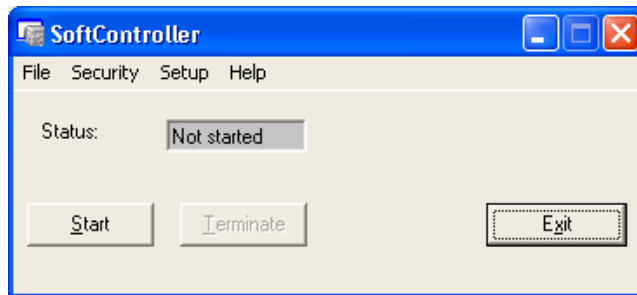


Figure 7. The SoftController start panel.

1. Click the **Start** button. The Status field displays *Started* and the SoftController starts.
2. Select **File > View log file**. A Session.LOG file will open in Note Pad.
3. Scroll down until you find a network address. An example of an IP address: 10.46.35.117:2.



*What is yours?*



To find out the IP address for a pc, simply open the command prompt (DOS editor) and write the command "ipconfig". Press Enter to view the IP address.

4. Close the Note Pad program.

You can learn more about running an application in a SoftController in [Section 5, Hardware Configuration](#) and [Section 6, Connecting the PLC and Go Online](#).

### Stopping the SoftController

5. Click the **Terminate** button. A SoftController dialog window opens.
6. Click **Yes**. The Status field displays *Not started* and the SoftController stops.
7. Click **Exit**.

## OPC Server

After OPC Server has been installed, it is easy to connect a controller to the OPC Server from the OPC panel. However, this requires that you first set up a System Identity from the Control Builder. Read how this is done in [Setting the System Identity in Control Builder](#) on page 89.

### Starting the OPC Server

Double-click the OPC Server icon on the desktop (if desktop shortcut is selected during installation), or from the Start menu on the Windows Task Bar,

**Start > All Programs > ABB Industrial IT PLC > AC 800M > OPC Server for AC 800M > OPC Server for AC 800M**

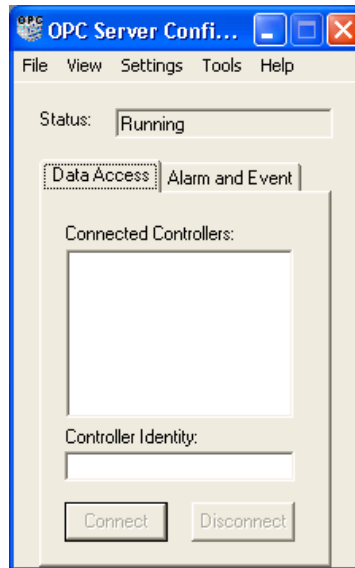


Figure 8. The OPC panel started after installation.

### Connecting the OPC Server

Make sure the Data Access tab is active:

1. Enter the IP address of the controller in the *Controller Identity* field and click the **Connect** button.
2. Select the Alarm and Event tab and repeat the step above.



How to create a PLC Id, see [Setting an IP Address](#) on page 84 and [Setting the System Identity in Control Builder](#) on page 89.

## Configuration Issues

The PLC Control Builder supports multi-user engineering, for more information see [Appendix A, PLC Control Builder AC 800M Settings](#).

This manual also covers configuration issues like upgrading projects to 4.0 versions and network redundancy. For more information regarding these issues see, [Appendix C, Upgrade](#) respective [Appendix B, Network Redundancy](#).

---

# Section 3 PLC Control Builder User Interface

## Introduction

This is a brief introduction to the PLC Control Builder and its core interface Project Explorer. Once familiarized with the Project Explorer, you are encouraged to study [Section 4, MyDoors Project](#), and build a Shop Door project.

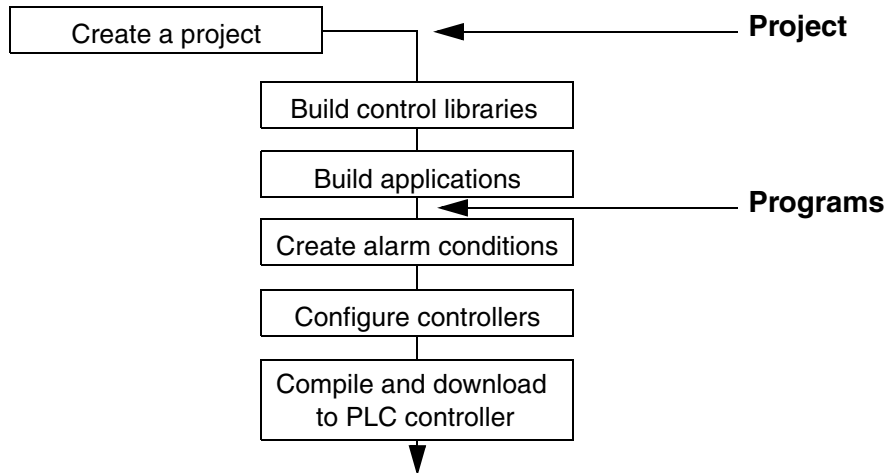
## About Programs and Projects

Engineers who are new to the Control Builder engineering environment, might think that a program and a project is the same thing.

It is important to learn the hierarchy used throughout PLC Control Builder. The following list below tries to describe the hierarchy in a descending order with start from the project level.

- 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 and additional objects (data types, function block types, control module types) that are used within the application.
- Each program is connected to a task, which decides how often the program is executed. It is also possible to connect individual function blocks and control modules to different tasks.

The sequence below in [Figure 9](#) tries to illustrate the steps from creating a new project to a download.



*Figure 9. Sequence for building a project and the hierarchy between a Project and Programs.*

## Project Templates

When you are about to create a new project, the Control Builder provides you with a set of predefined templates, typical for a control system. The templates contain predefined initial setup data, suitable for different kind of projects. The PLC Control Builder provides you with the following project templates:

- AC800M
  - (Normal use)
- SoftController
  - (Development use, without access to a controller)
- EmptyProject
  - (Rare use, a minimum configuration with only the System folder inserted)

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

## Project Explorer

Project Explorer is the “core” user interface to the Control Builder programming tool. It displays the currently active project. Only one project can be open at the same time.

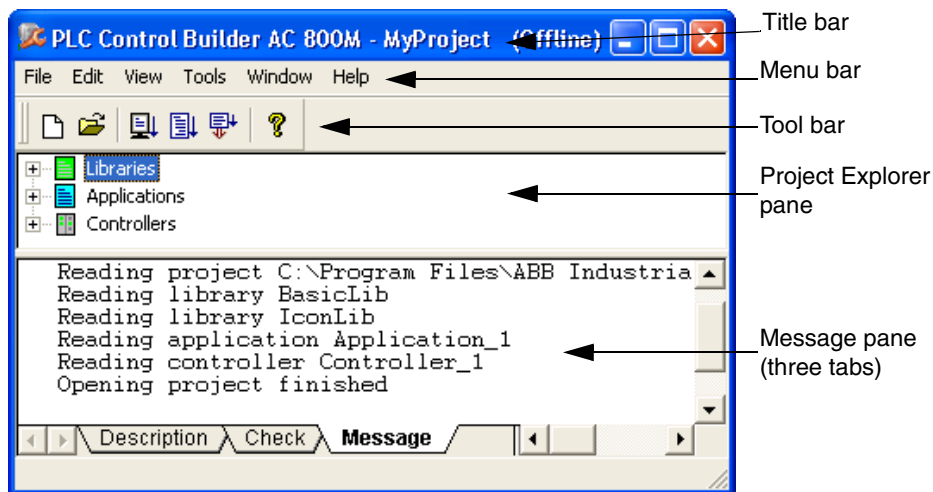


Figure 10. Project Explorer.

### Title Bar, Menu Bar and Tool Bar

The title bar shows the project name (for example MyProject in [Figure 10](#)).

The menu bar contains the drop-down menus File, Edit, View, Tools, Window and Help. When 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 online help and download.



For detailed information of the content of menus and tool bar, please refer to Control Builder online help.

## Project Explorer Pane

The Project Explorer pane contains three main folders, see [Figure 11](#):

- The Libraries folder, see [Libraries Folder](#) on page 40.
- The Applications folder, see [Applications Folder](#) on page 40.
- The Controllers folder, see [Controllers Folder](#) on page 41.

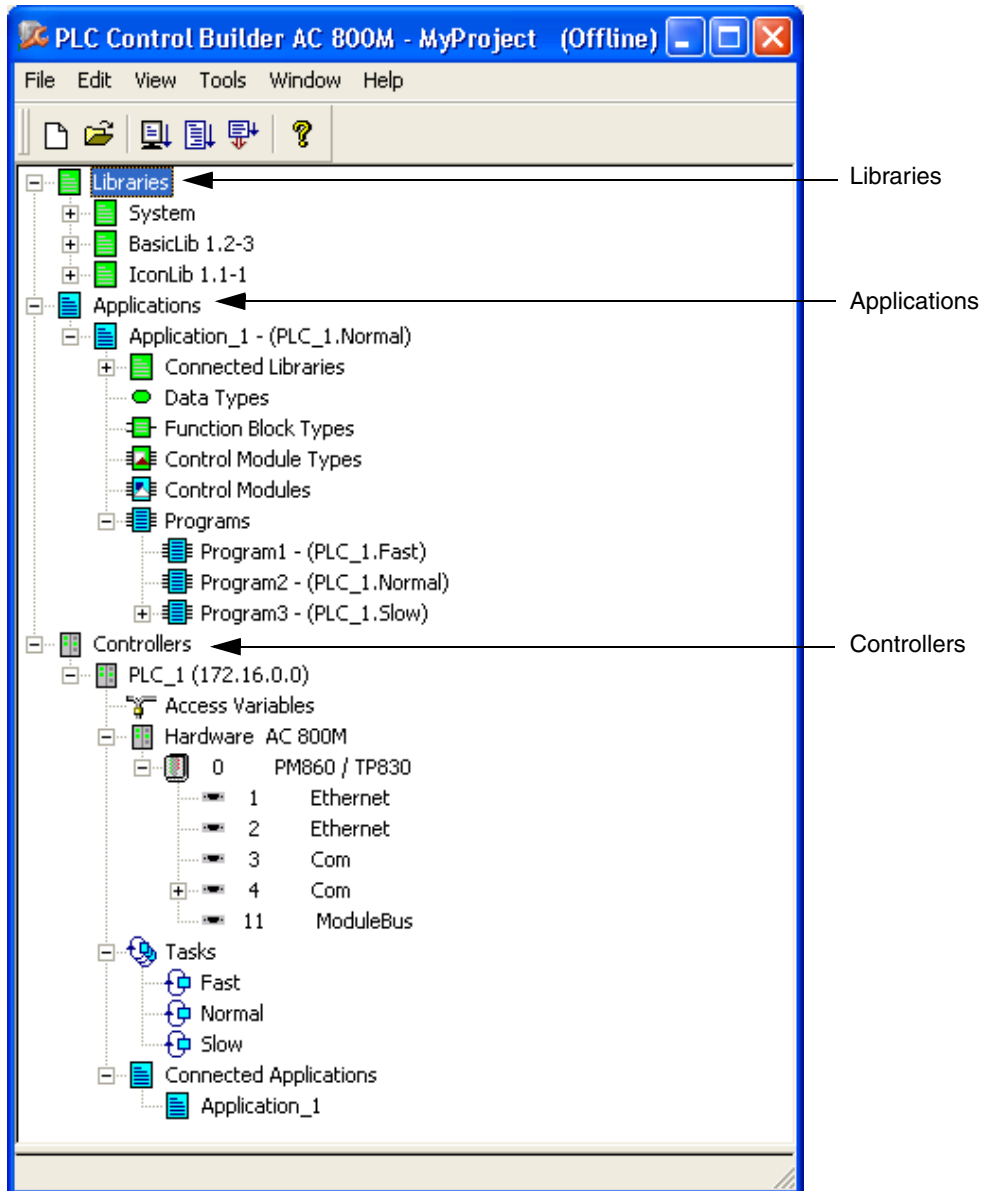


Figure 11. The Project Explorer pane, showing the three main folders, Libraries, Applications, and Controllers.

## Libraries Folder

When a project is created for AC 800M, the Libraries folder contains a System folder (firmware functions) and two libraries, the Basic library and a supportive Icon library. These libraries are automatically connected to the default Application\_1. However, as you build your projects, you may choose to either insert standard libraries or create self-defined libraries.

Additional libraries can always be added to an application as long as it has been inserted into the Libraries folder.



For more information on libraries and library handling, see Online Help.

## Applications Folder

The Applications folder holds all code that is (or is about to be) downloaded to the controller(s). This code can either be stored as programs, or as control module types or single control modules. Which you choose depends on the requirements of your particular application.

The Connected Libraries folder contains all libraries that are connected to this particular application. Libraries are connected by right-clicking this folder and selecting Connect Library. However, only libraries that have already been inserted to the project can be connected to an application. In order to access the types inside a library; it must be connected to the application. Connect a library to an application by simply right-clicking the Connected Libraries icon and select a library from the drop-down menu.

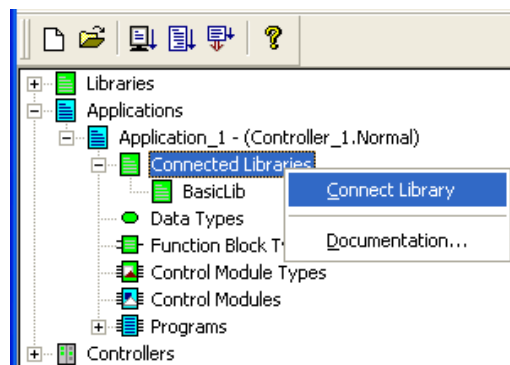


Figure 12. The context menu for connecting a library to an application.



The Application folder contains three sub-folders for collecting types: Data Types, Function Block Types, and Control Module Types. You can either insert an existing type (from another project) or create a new type within any of these three sub-folders.

There are two ways of organizing your code, in programs or in control modules. This is reflected by the two folders Control Modules and Programs. Control modules are stored in the Control Modules folder, while programs are stored in the Programs folder. By default, the Programs folder contains three programs. Each of these three programs are connected to a default task. You can change the task connections, as well as add your own tasks and programs.



From the objects in the Applications folder, a number of software editors can be opened, see [Editors](#) on page 44.

You can check your code for errors by clicking on the Check icon on the tool bar. Errors are indicated by a red triangle next to the object in question (in offline mode). Descriptions of the errors (if any) are displayed in the Check tab of the message pane.

## Controllers Folder

The Controllers folder contains all PLCs that belong to the project. For each PLC, there is an Access Variables container and a CPU unit icon to which other hardware units, such as I/O units and communication interfaces can be added. Icons can also be added to the PLC on the same level as the CPU unit. The structure inside the Controllers folder is meant to mirror the physical structure, which means that all ports and buses have their own corresponding icon in Project Explorer.



For more information about hardware configuration and the Controllers folder, see [Configure Hardware](#) on page 73.

The Controllers folder also contains a sub-folder Tasks. 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. The Connected Applications folder contains all the applications that are connected to the PLC.

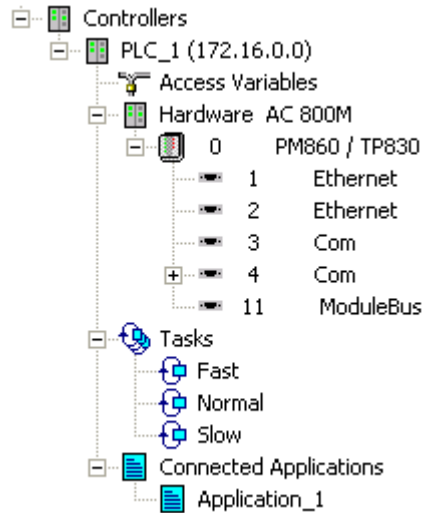


Figure 13. The controller structure in Project Explorer, with corresponding icons for ports and buses.

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 44.

### Context Menu

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

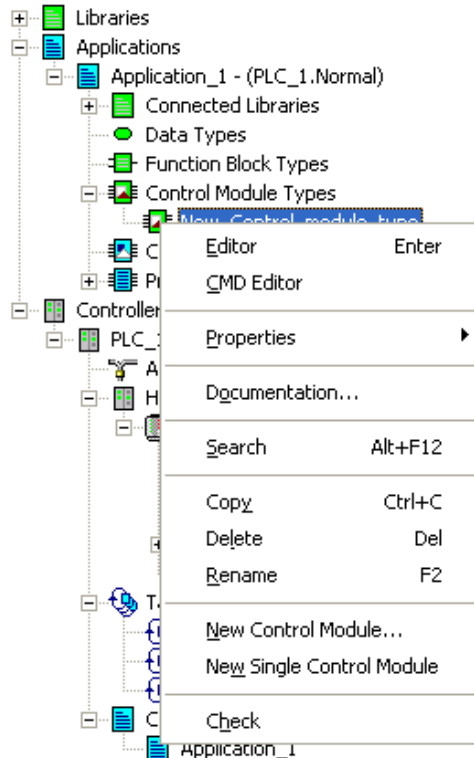


Figure 14. Context menu in Project Explorer.

### Message Pane

The message pane (see [Figure 6](#)) 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. To access an editor, right-click the object (it could be a controller, another hardware unit, an application, a program, or a type) and select the editor from the context menu.

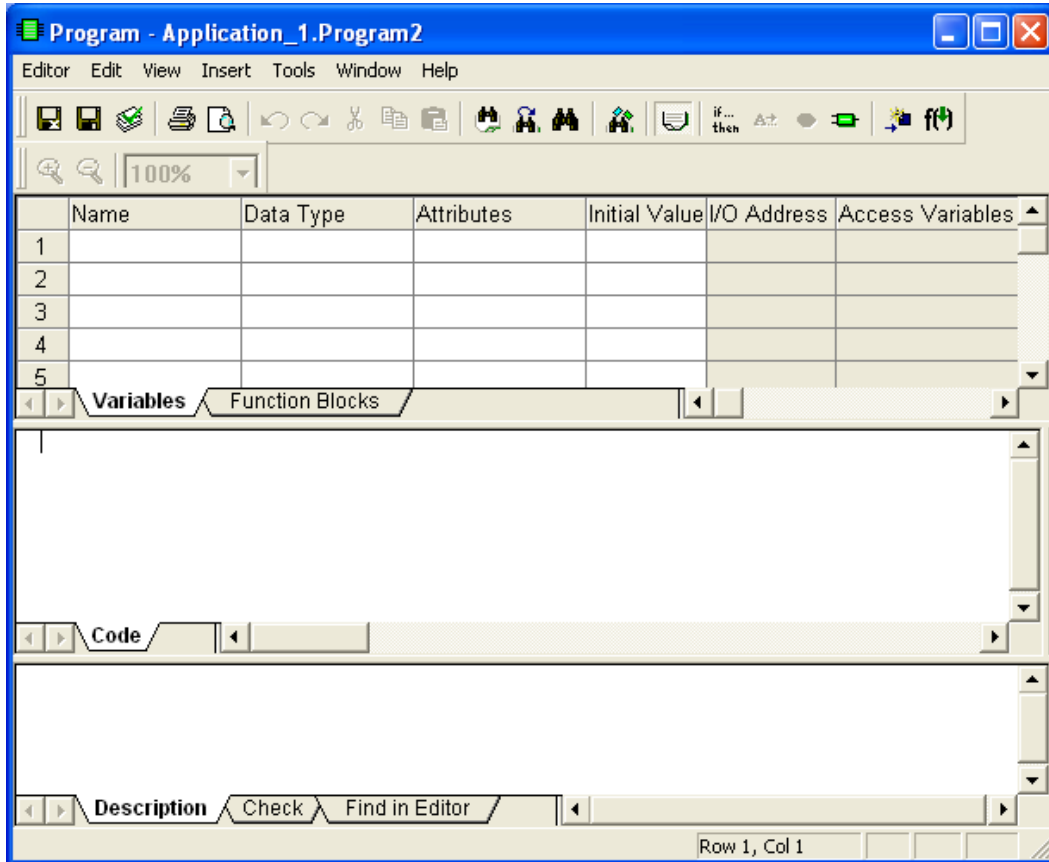


Figure 15. Program editor.

Among many things, editors are used to declare project constants, and parameters, as well as to declare variables and connect them 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.

---

## Section 4 MyDoors Project

This section encourages you to build a small project and getting yourself familiarized with the Control Builder environment. 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 how to declare variables, function blocks and separating code by using code blocks, and much more.

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 program execution.



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

However, if you by any chance need to study the MyDoors project example (immediately), the Control Builder installation comes with an already made project example for you, called *ShopDoors*. See the subsection [Project Examples in Control Builder](#) on page 69, for locating the project example.



For locating the ShopDoors example, or any other Control Builder examples, simply follow the instruction given under section [Project Examples in Control Builder](#) on page 69.

After finishing your study of the MyDoors project you are advised to carry on with the next couple of sections:

- [Section 5, Hardware Configuration](#) which covers how to setup a hardware configuration according to your control system.
- [Section 6, Connecting the PLC and Go Online](#) covers all the essential steps for making it possible to download an application and go online.

## MyDoors Project

Before you create your project and start writing code, take a brief moment and study the Specifications below and the [Defined Variables](#) on page 47.

### 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. 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

### Creating a New Project

1. From the Project Explorer, select **File > New > Project**. A New Project window opens.

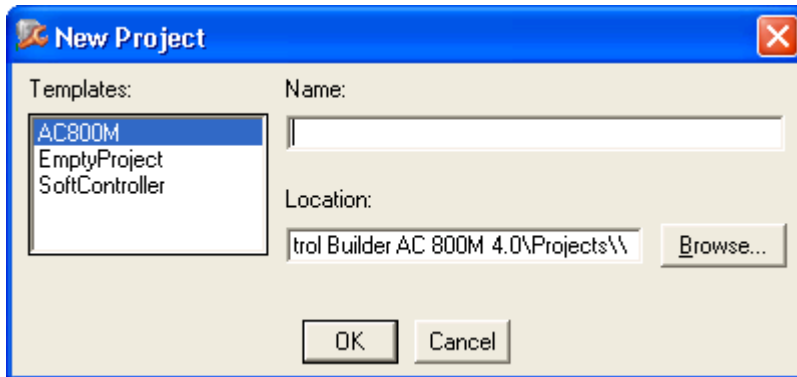


Figure 16. The New Project window for setting up a project.

2. Select the **AC800M** template and type *MyDoors* in the **Name** field. Ignore the given location path (for now).
3. Click **OK**. Project Explorer creates and opens MyDoors project, see [Figure 17](#).



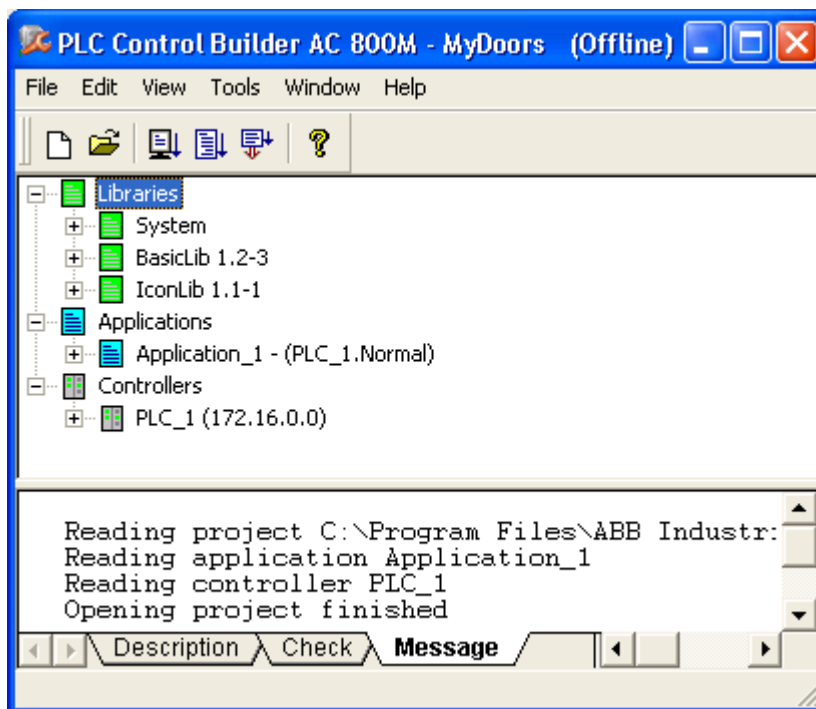


Figure 17. MyDoors project opened in Project Explorer.

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



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.

## Local Variables

There are different types of variables in PLC Control Builder for storing and computing values (local, global, and access variables), where the local variables are the most frequently used in Control Builder. They always belong to the local code inside a function block, control module or a program.

In this example, you will declare local variables in a program named **Program2**.

### Declaring Local Variables

1. In the Project Explorer, expand the project tree until you see **Program2**, (Figure 18). Double-click the icon to open the Program editor.

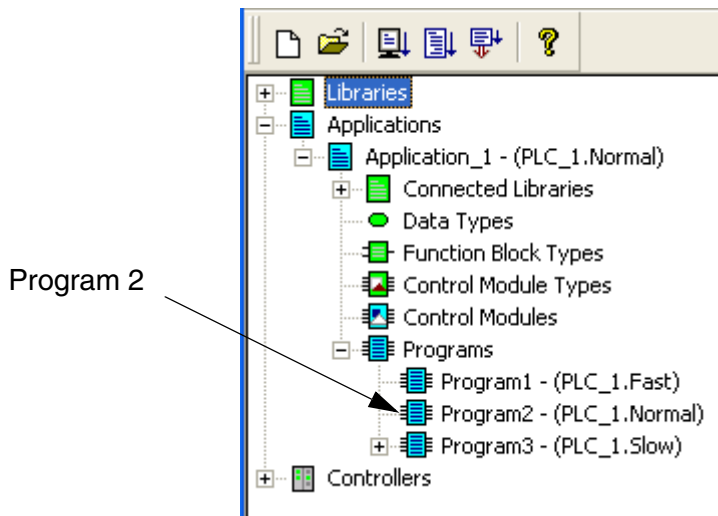


Figure 18. Programs folder expanded with three preset programs available.

2. The program editor is divided into three panes: the declaration pane, the code pane, and the message pane. (See Figure 19.)

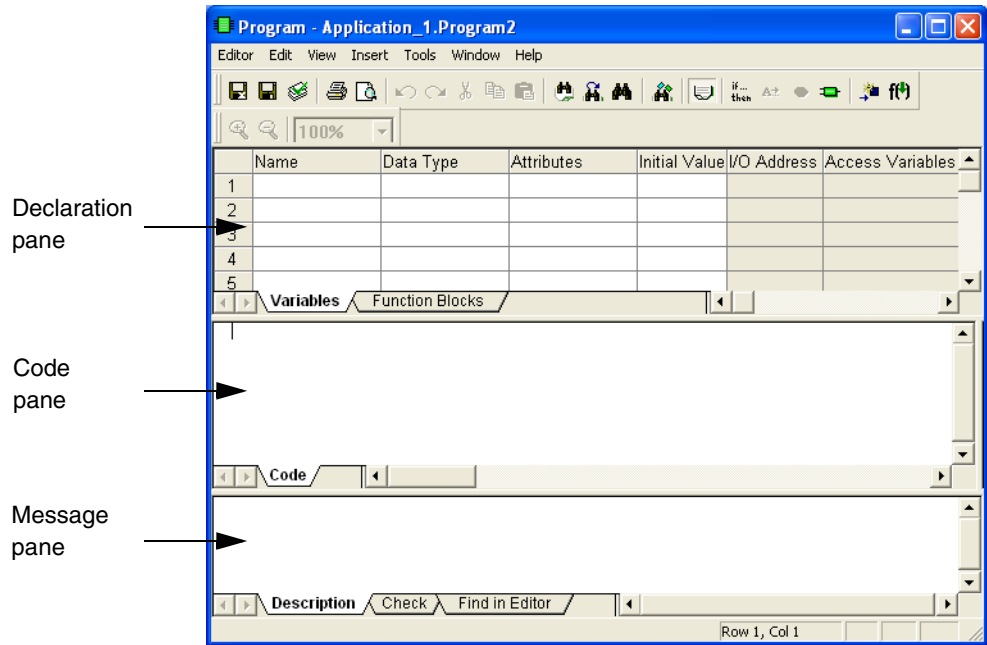


Figure 19. The editor for program *Program2*.

3. 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 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 configuring the hardware.
8. The last column ‘Description’ is reserved for you to use freely. Your first row should now look like row 1 in Figure 20.

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

Figure 20. 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 21](#).

	Name	Data Type	Attributes	Initial Value	I/O Address	Access Variables	Description
1	Photo_Cell	bool	retain	false			Photo cell to doors
2	DoorsOpen_Time	time	constant	T#5s			Time duration that doors should be opened
3	DoorsOpen_ET	time	retain				Elapsed time after photo cell has been activated

*Figure 21. Declaration of the `DoorsOpen_Time` and `DoorsOpen_ET` variables.*



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. Go ahead and try!



10. Declare a variable named, `DoorsOpen_ET` that records the time elapsed since the photocell was activated last time. Complete the declaration of `DoorsOpen_ET` according to row 3 in [Figure 22](#).
11. Declare the remaining variables (with start from row 4) in the grid according to [Figure 22](#).

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	Service_Req	bool	retain	false			Flag that is set when service is required
10	Customers_Qty	dint	retain	0			Total number of customers
11	Reset_Counter	bool	retain	false			Flag that resets the number of customers

*Figure 22. Declaration of the remaining variables.*



For data types that can be specified in the Data type column, choose **Insert > Variable, Type, Attribute** (or press CTRL+J) to access a list of possible data types.

12. Click **Check**  to check for errors.
13. Click **Save**  to save the variables.

## Function Blocks

Timers and counters in the PLC 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

Make sure the program editor is open, (see [Figure 18](#), to open editor)

1. Select the **Function Blocks** tab in the declaration pane.
2. Place the cursor in the upper left-hand cell in the declaration pane and type OpenDoors.
3. Move one cell to the right by pressing the tab key. Right-click the cell and select **Insert > Variable, Type, Attribute** from the context menu. A dialog list opens.

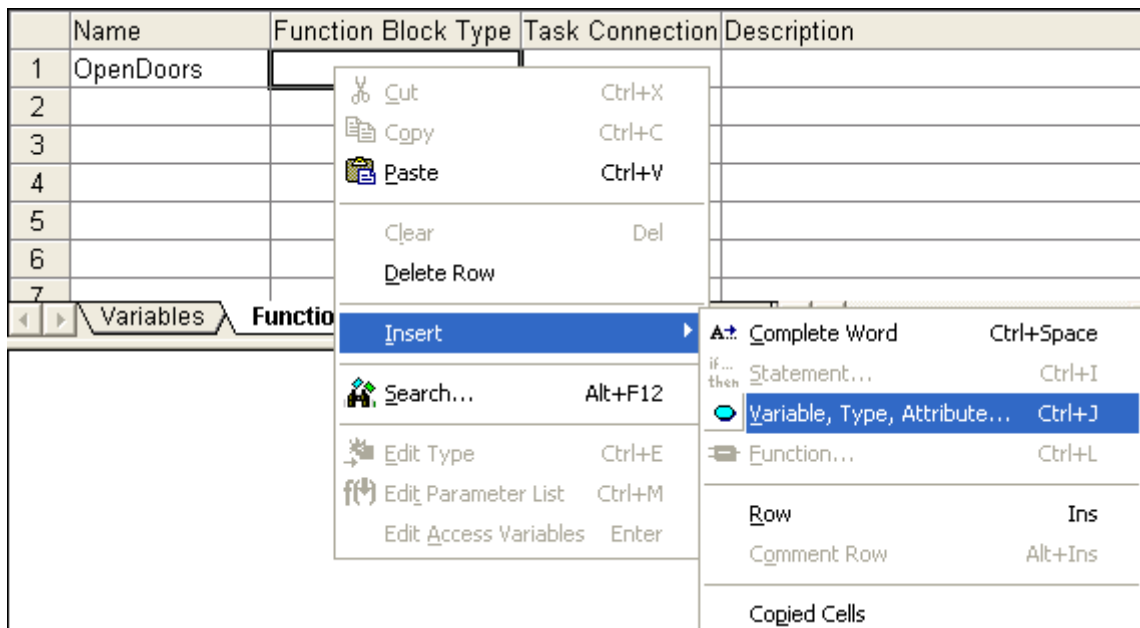


Figure 23. The path in the context menu for selecting (for example) function blocks.

4. Type **TO** (or TOF) to jump down to the TOF Function Block type in the list.

	Name	Function Block Type	Task Connection	Description
1	OpenDoors	TOF		
2		TimerU		
3		TOF		
4		TOOn		
5		TP		
6		Trigger		
7		BasicLib 1.2-3		

Variables Function Blocks

Figure 24. The TOF function block type is selected.

5. Press the ENTER key. The TOF is declared in the program editor. Write Description text according to Figure 25.
6. Declare two CTU function blocks in row 2 and row 3 in the same way as you did with the TOF function block. Name them Customer\_Count\_Up and Service\_Count\_Doors, according to Figure 25.

	Name	Function Block Type	Task Connection	Description
1	OpenDoors	TOF		Timer for motor
2	Customer_Count_Up	CTU		Counter for the number of customers
3	Service_Count_Doors	CTU		Counter for the number of openings o doors

Variables Function Blocks

Figure 25. Declaring the function blocks.



For 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**.

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

## Code Blocks

Both programming editors (programs and control modules) support code blocks. All the code blocks with the exception of the first one are always self-defined which means that you create your own code blocks for structuring code. Thus, code blocks are a way of enhancing the readability, traceability, and structure, of your programming code. However, you should always strive to keep the number of code blocks to a minimum, or you might risk ending up with a badly arranged programming structure.

The code blocks are always executed in a predetermined order, and in this example from left to right (programs).



Code block names cannot contain certain characters. See online help for information on which characters that cannot be used in code block names.

### Creating Code Blocks

1. Right-click the Code tab and select **Rename** from the context menu. A Rename Code Block window opens.

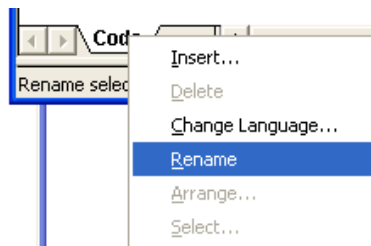
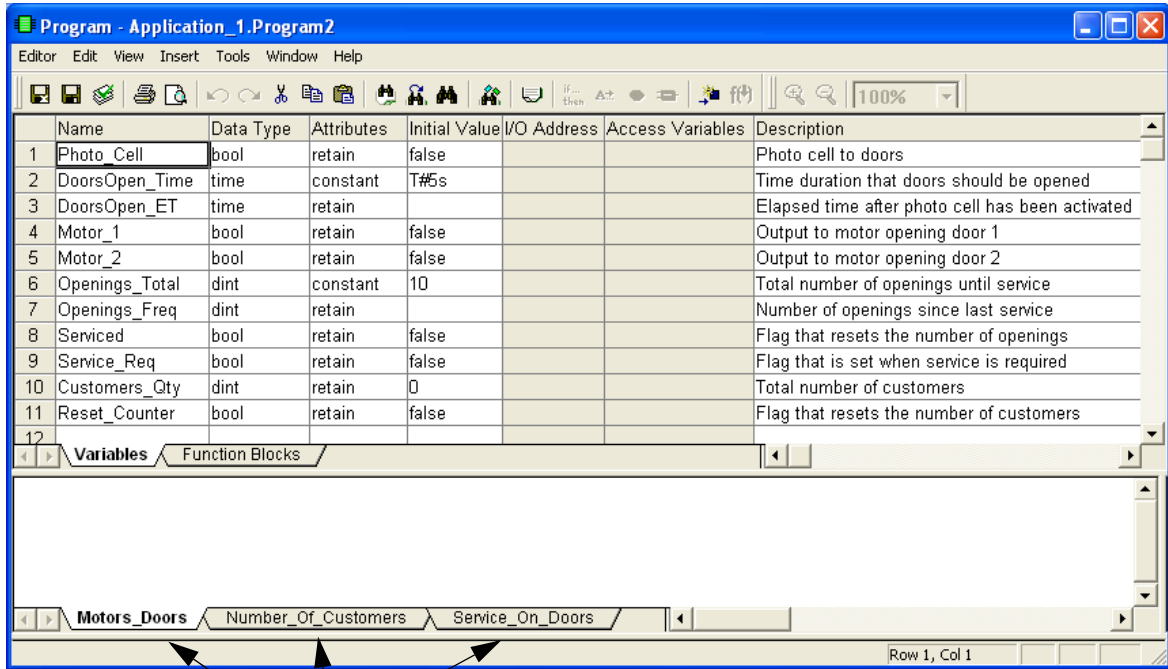


Figure 26. Right-click the code block tab to access the context menu.

2. Write `Motors_Doors` in the Name field.
3. Click **OK**.
4. Right-click the `Motors_Doors` tab and select **Insert** from the context menu. An Insert New Code Block window opens.
5. Write `Number_Of_Customers` in the Name field and check that the Structured Text language is selected. Click **OK**.

6. Add a third code block in the same way and name it `Service_On_Doors`.



Code Blocks

Figure 27. The program editor with three new code blocks created.

After creating the three code blocks representing motors, customers and service issues in your program editor, you are now ready to write corresponding code block specific programming code.



The code blocks are executed from left to right, thus first `Motors_Doors`, then `Number_Of_Customers` and finally `Service_On_Doors`.



## Code Input

### Making a Function Block Call in Motors\_Doors

1. Select the code block tab **Motors\_Doors** and place the cursor in the code pane.
2. Select **Insert > Variable, Type, Attribute** from the Menu bar (or press CTRL+J). A dialog list opens.



The variables and function blocks declared under the **Function Blocks** tab in the declaration pane will be shown in the dialog list.

3. Select the **OpenDoors** function block in the list and press the ENTER key.



Start typing the beginning letters in OpenDoors (i.e. **Op...**) and you will automatically land on OpenDoors in the scroll list.

4. Make sure the cursor is located directly after `OpenDoors` in the code pane. Write a left-hand parenthesis '`(`'.



Write a left-hand parenthesis



Figure 28. Write a left-hand parenthesis after `OpenDoors` in the code pane.

- When you type the leading left-hand parenthesis ‘ ( ‘, a Function block call editor will open, see [Figure 29](#).

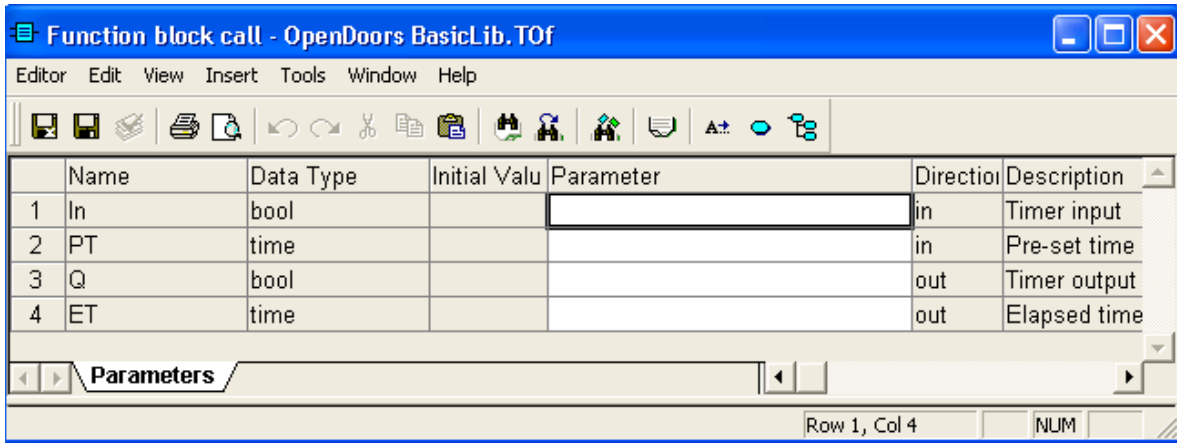



Figure 29. The Function block call editor.

- Place the cursor in the first empty white cell and select **Insert > Parameter From List** (or press CTRL+J, or click  -icon in the Menu bar). A variable list opens.
- Select **Photo\_Cell** and press the ENTER key to accept the selection.

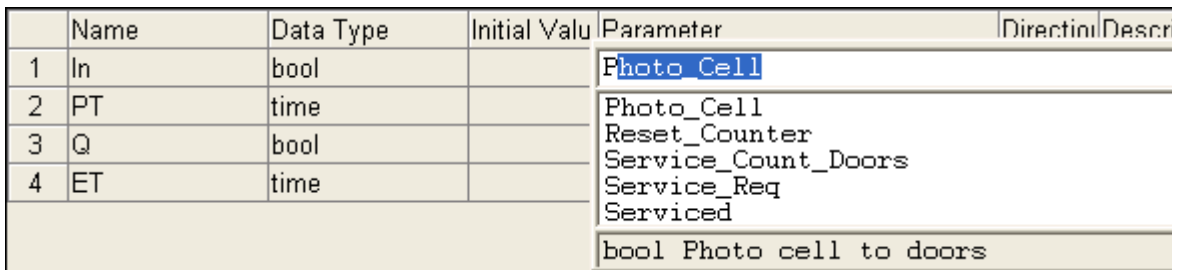


Figure 30. Variable list in the Function block call dialog.

8. Fill in the other two variables in the parameter list according to [Figure 31](#).

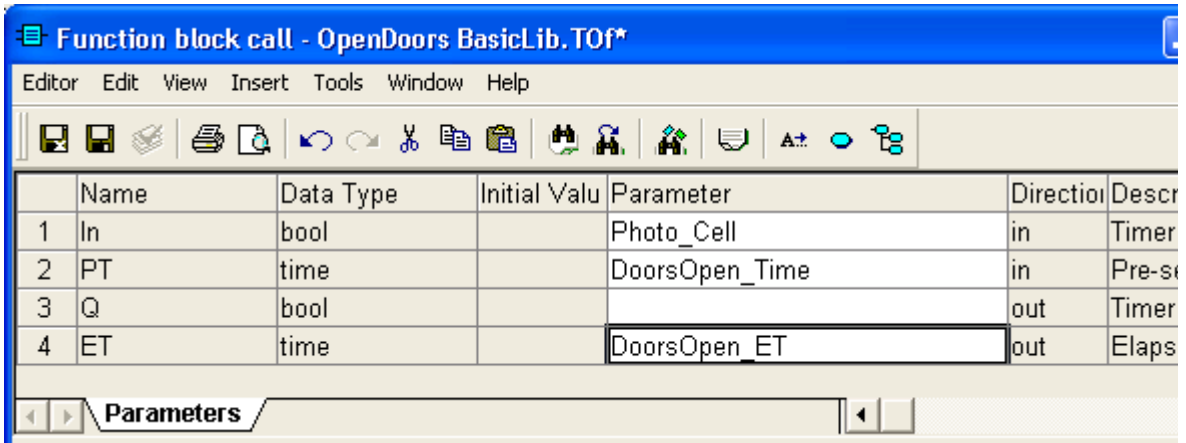



Figure 31. Connecting function block parameters.

9. Click **Save and Close**  to insert the parameters into the code.

### 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. To achieve this, write the following code in the code pane:

```
Motor_1 := OpenDoors.Q;
Motor_2 := OpenDoors.Q;
```

The output Q is now addressed directly to the function block and a value assigned to both motors to open the doors. See [Figure 32](#).

```

OpenDoors( In := Photo_Cell,
           PT := DoorsOpen_Time,
           ET => DoorsOpen_ET );

Motor_1 := OpenDoors.Q;
Motor_2 := OpenDoors.Q;

```

Figure 32. The code block *Motors\_Doors*.



The code can be written structured with or without tabs and spaces.

10. Click **Save** to save the code.

### Making a Function Block Call in *Number\_Of\_Customers*

1. Select the code block tab *Number\_Of\_Customers* and place the cursor on the first line in the code pane.
2. Select **Insert > Variable, Type, Attribute** from the Menu bar (or press CTRL+J). A dialog list opens.



The variables and function blocks declared under the **Function Blocks** tab in the declaration pane will be shown in the dialog list.

3. Select the *Customer\_Count\_Up* function block in the list.



Start typing the beginning letters in *Customer\_Count\_Up* (i.e. **Cu...**) and you will automatically land on *Customer\_Count\_Up* in the scroll list.

4. Accept the selection by pressing the ENTER key. Type the leading left-hand parenthesis ‘(’. A Function block call editor opens.
5. Place the cursor in the first empty white cell and select **Insert > Parameter From List** (or press CTRL+J). A variable list opens.

- Select Photo\_Cell in the list and press the ENTER key to accept the selection.

	Name	Data Type	Initial Value	Parameter	Direction	Description
1	CU	bool		Photo_Cell		
2	Reset	bool		Photo_Cell		
3	PV	dint		Reset_Counter		
4	Q	bool		Service_Count_Doors		
5	CV	dint		Service_Req		
				Serviced		
				bool Photo cell to doors		

- Fill in the other two variables in the parameter list according to Figure 33, thus leave out the connection of the two parameters PV and Q.

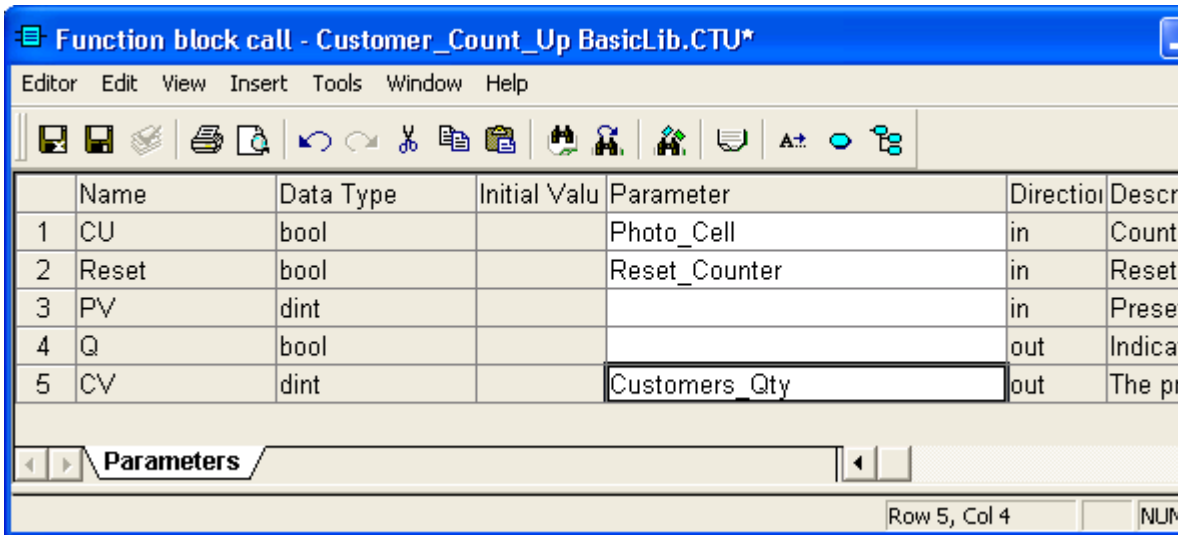



Figure 33. Connecting CTU function block parameters.

- Click **Save and Close**  to insert the parameters into the code. See Figure 34.

```

Customer_Count_Up( CU := Photo_Cell,
                  Reset := Reset_Counter,
                  CV => Customers_Qty );

```

The screenshot shows a code editor window with a tab labeled 'Number\_Of\_Customers'. The code block contains the following text: `Customer_Count_Up( CU := Photo_Cell, Reset := Reset_Counter, CV => Customers_Qty );`. The status bar at the bottom indicates 'Row 1, Col 2' and 'NUM'.

Figure 34. The code block `Number_Of_Customers`.

### Making a Function Block Call in `Service_On_Doors`

1. Select the code block tab `Service_On_Doors` and place the cursor on the first line in the code pane.
2. Select **Insert > Variable, Type, Attribute** from the Menu bar (or press CTRL+J). A dialog list opens.



The variables and function blocks declared under the **Function Blocks** tab in the declaration pane will be shown in the dialog list.

3. Select the `Service_Count_Doors` function block in the list.
4. Accept the selection by pressing the ENTER key. When you now type the leading left-hand parenthesis '(', a parameter editor will open.

The screenshot shows a dialog box titled 'Function block call - Service\_Count\_Doors BasicLib.CTU'. It has a menu bar (Editor, Edit, View, Insert, Tools, Window, Help) and a toolbar with various icons. Below the toolbar is a table with the following data:

	Name	Data Type	Initial Valu	Parameter	Direction	Desc
1	CU	bool			in	Coun
2	Reset	bool			in	Rese
3	PV	dint			in	Pres
4	Q	bool			out	Indica
5	CV	dint			out	The p

At the bottom of the dialog, there is a 'Parameters' tab and a small input field.

5. Place the cursor in the first empty white cell and select **Insert > Parameter From List** (or press CTRL+J). A variable list opens.
6. Select `Motor_1` in the list and press the ENTER key to accept the selection.

	Name	Data Type	Initial Value	Parameter	Direction	Description
1	CU	bool		Motor_1		
2	Reset	bool		Motor_1		
3	PV	dint		Motor_2		
4	Q	bool		OpenDoors		
5	CV	dint		Openings_Freq		
				Openings_Total		
				bool		

7. Fill in the other variables in the parameter list according to [Figure 35](#).

	Name	Data Type	Initial Value	Parameter
1	CU	bool		Motor_1
2	Reset	bool		Serviced
3	PV	dint		Openings_Total
4	Q	bool		Service_Req
5	CV	dint		Openings_Freq

Figure 35. Connecting CTU function block parameters.

8. Click **Save and Close**  to insert the parameters into the code. See [Figure 36](#).

```

Service_Count_Doors( CU := Motor_1,
                    Reset := Serviced,
                    PV := Openings_Total,
                    Q => Service_Req,
                    CV => Openings_Freq );
    
```



Motors\_Doors
Number\_Of\_Customers
**Service\_On\_Doors**

Figure 36. The parameters connected in the code block, `Service_On_Doors`.



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 PLC 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 a PLC.

The test mode is an easy way to try out the application many time. 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.



Before you can run the program in Test mode, a Difference Report window will open. However, the Difference Report function is not important for this example since it does not generate a report in Test mode. Choose to keep the default setting, or read how to turn off the function in the subsection [Disable/Enable Difference Report](#) on page 116. This example assumes that the Difference Report has the default setting enabled.

1. In Project Explorer, click **Test Mode** . The Test Mode Analysis window opens.
2. Click **Cold Restart All**.
3. Click **Continue**. A Difference Report window opens.



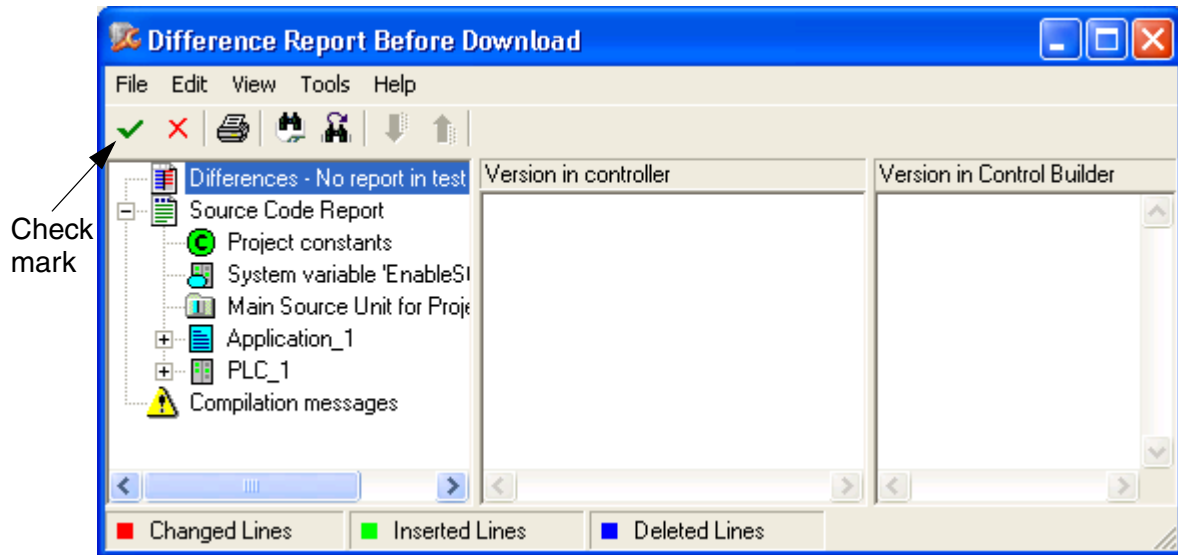


Figure 37. Difference Report Window.

4. Select the green check mark (see [Figure 37](#)) to continue. Difference report is not generated in Test mode.
5. Double-click **Program2** to display the editor.
6. Select **Motors\_Doors** tab. All variables in **Program2** are listed in the upper pane and the code in the lower pane, see [Figure 38](#).

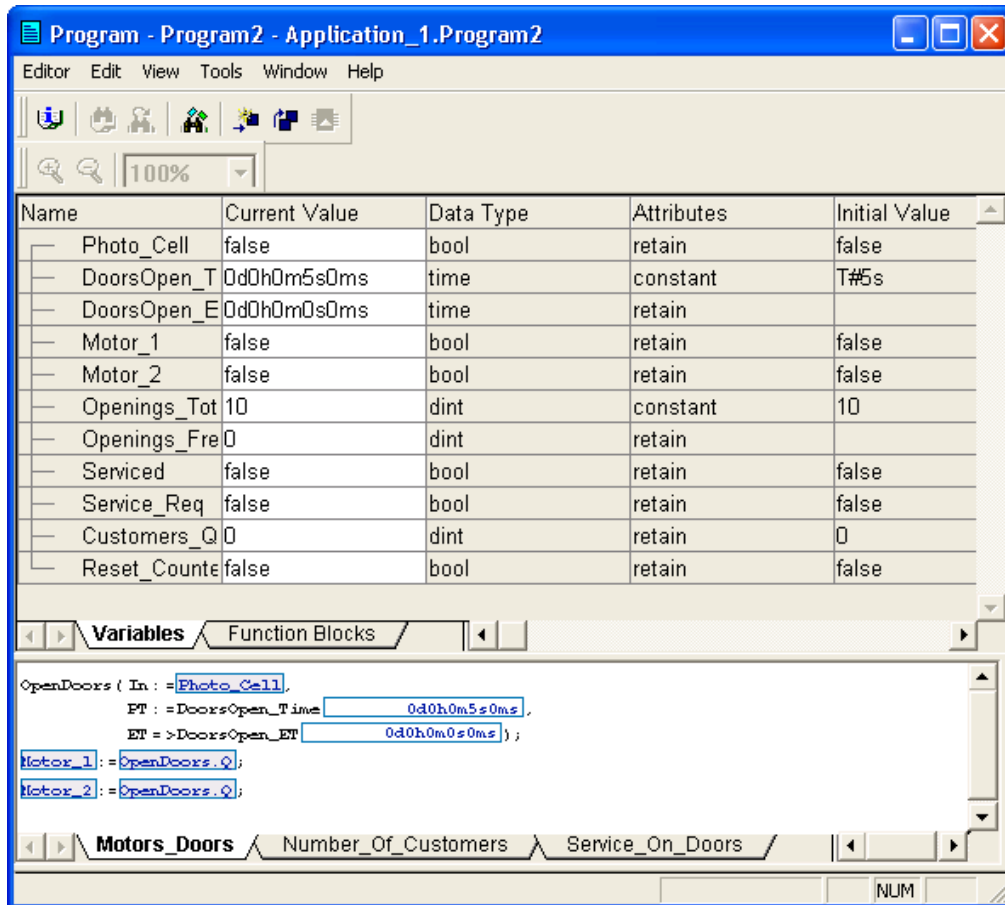


Figure 38. The Program2 editor in Test mode.

### Analyzing the Code During Program Executions

As you can see, test mode helps you test and analyze your project without yet having a PLC ready in the Project Explorer tree. You can change the variable values and study the program response.

While analyzing the variable conditions, the following instructions will ask you to right-click a variable. These can be selected from either the parameter list or directly in the code pane.

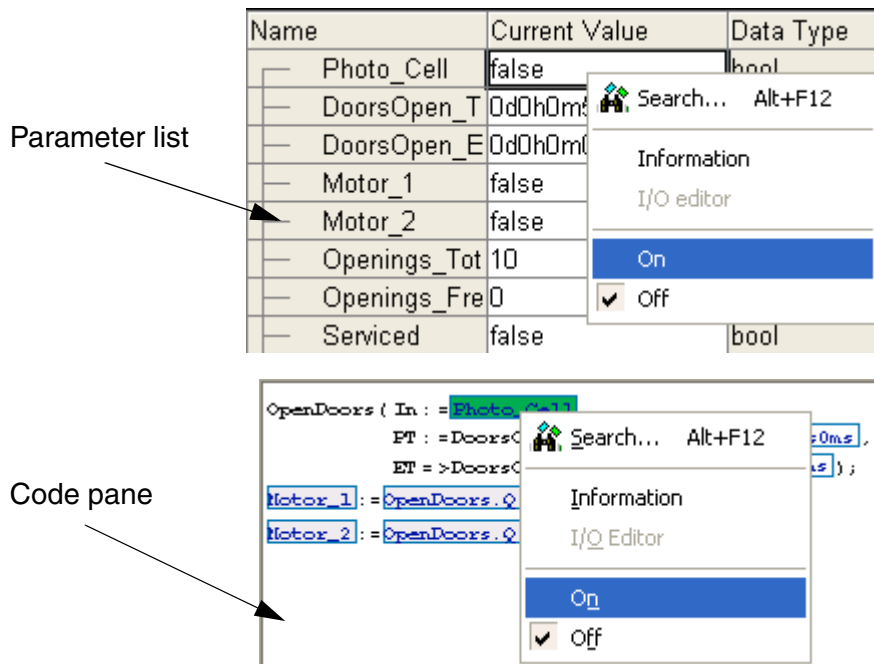


Figure 39. Changing the current value on a variable from the Parameter list, or in the code pane.

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 now have three openings of the doors and five customers.

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

## Project Examples in Control Builder

All the examples are Read-only, which means they cannot be altered. If you only wish to study (read) them, they can be opened as they are, see [Reading the ShopDoors Example](#) on page 71. However, if you wish to alter something in the example, like using it as a template or a skeleton for other projects, then you must first prepare the project example, as described below.

### Preparing the ShopDoors Example for a Project

From the disk where PLC Control Builder<sup>1</sup> is installed:

1. Start the Search tool in Windows and search for *ShopDoors\_ST.prj*. The project file will be located inside the example folder for the ShopDoors project.
2. Copy the complete **ShopDoors\_ST** folder into the **Projects** folder<sup>2</sup>.
3. Right-click the **ShopDoors\_ST** folder and select **Properties** from the context menu.
4. Clear the folders properties Attribute Read-only. Apply changes to the folder, sub folder and files.
5. Done! Follow the next steps under [Opening a Project](#) on page 70, if you wish to open the now converted ShopDoors\_ST project.

---

1. PLC Control Builder is normally installed on the Local Disk (C:)

2. The Projects folder is normally located at; C:\ABB Industrial IT Data\Engineer IT Data\PLC Control Builder AC 800M 4.0\Projects

## Opening a Project

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

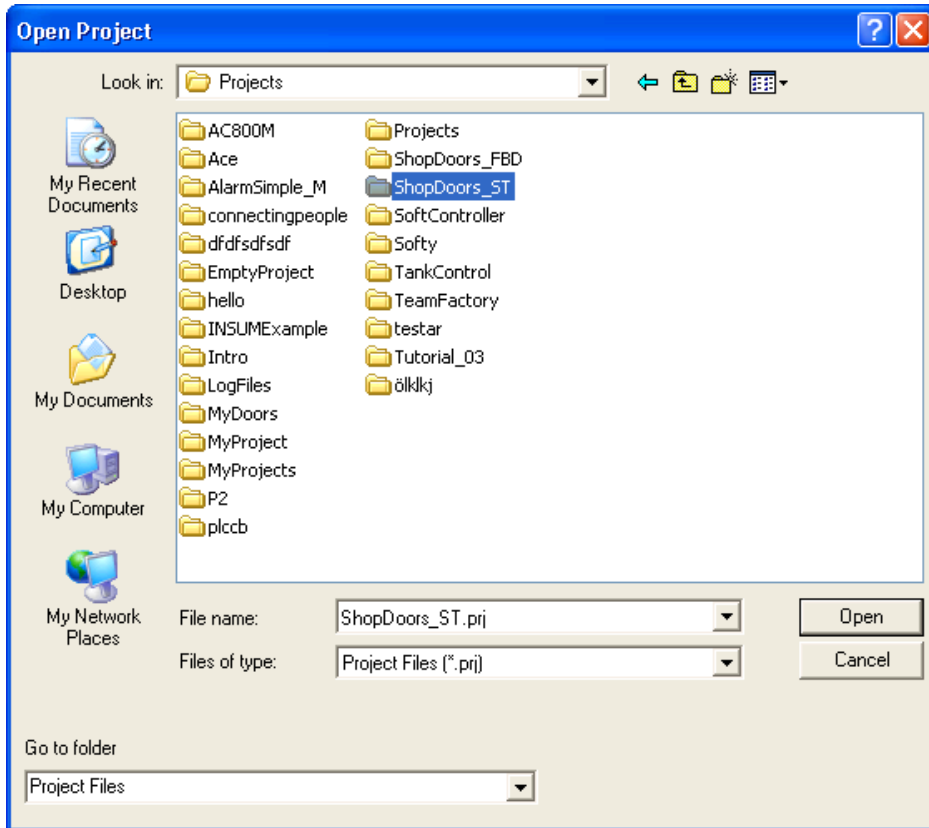


Figure 40. The Open project window.

2. Select the **ShopDoors\_ST** folder and click the **Open** button. The project ShopDoors\_ST folder opens.
3. Select the **ShopDoors\_ST.prj** file and click the **Open** button. The Project Explorer opens the ShopDoors\_ST project.

### Reading the ShopDoors Example

1. From the Project Explorer, select **File > Open Project**. An Open Project window opens (Figure 40).
2. Press **Go to folder** drop-down menu and select **Example files**. The Example folder opens.

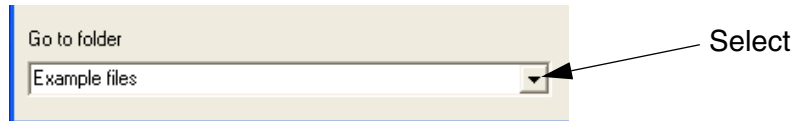


Figure 41. Drop-down menu for selecting project files or example files.

3. Open the **ShopDoors\_ST** folder and select the **ShopDoors\_ST.prj** file.

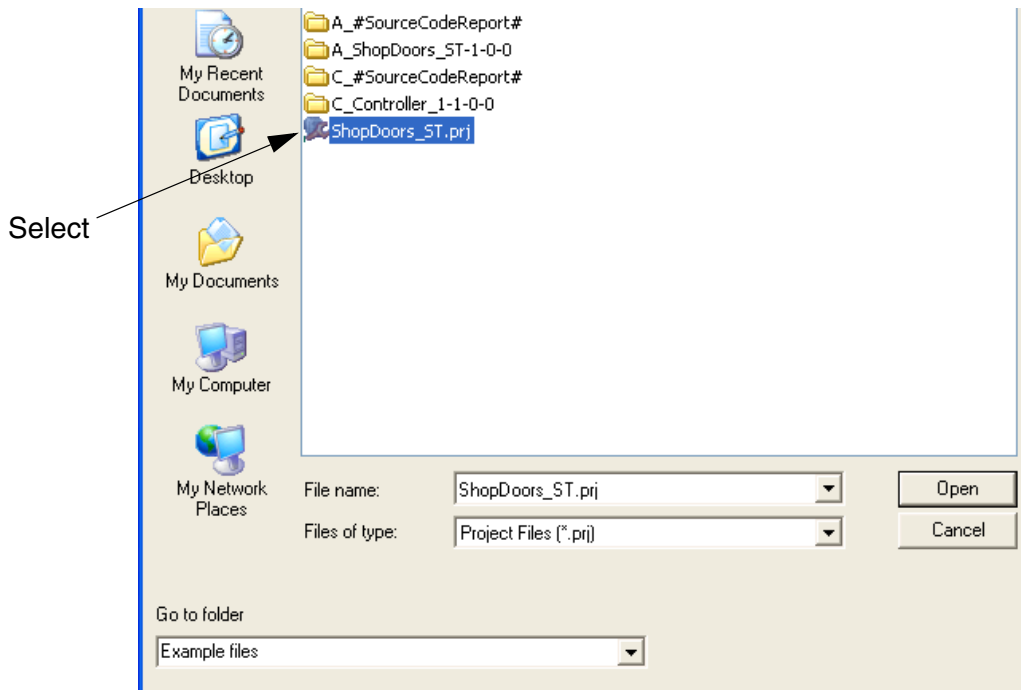


Figure 42. The Open project window, showing the contents of the ShopDoors\_ST directory. Note the (ShoopDoors\_ST.prj) project files.

4. Click **Open**. The ready-made ShopDoors example will open in Project Explorer.





---

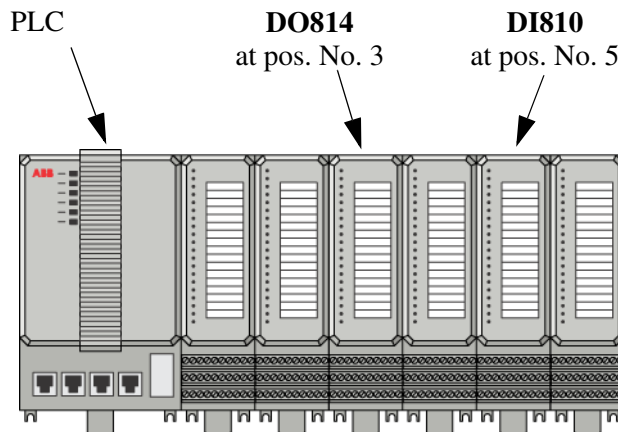
## Section 5 Hardware Configuration

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.

The Controller object in Project Explorer is the root to which hardware objects are added.

### Configure Hardware

Study the hardware configuration in [Figure 43](#). Assume a PLC, together with six I/O modules. We are going to add two of them to the tree structure in Project Explorer. Add DO814 and DI810, at positions 3 respective 5.



*Figure 43. Hardware position for IO modules (for example DO814 at position 3 and DI810 at position 5).*

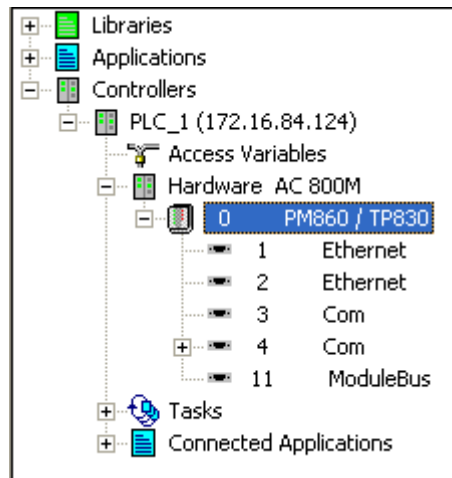
### Adding the IO Modules DO814 and DI810

1. Expand **Controllers > PLC\_1 > Hardware AC 800M > PM860/TP830**, in the Project Explorer tree.

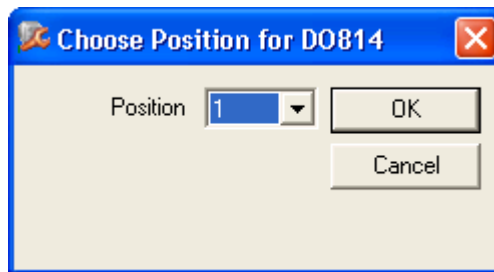


You must have the correct CPU unit connected in your hardware tree; otherwise you cannot download your application to the PLC. If you run with a **SoftController** the choice of CPU models is optional.

- To Replace a CPU: Right-click the **PM860** item and select **Replace With** in the context menu.



2. Right-click the **ModuleBus** item and select **New Unit > S 800I/O > DO814**. A Choose Position for DO814 window opens.



3. Select position **3** from drop-down menu and click **OK**.

4. Right-click the **ModuleBus** item and select **New Unit > S 800I/O > DI810**. A Choose Position for DI810 window opens.
5. Select position **5** from the drop-down menu.
6. Click **OK**.

When you have added the two IO modules, your “hardware tree” should look like [Figure 44](#).

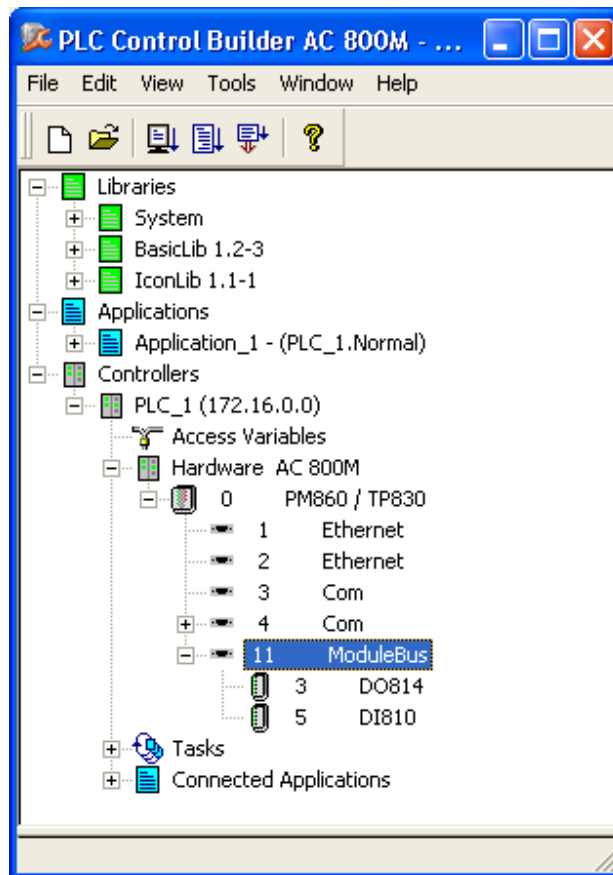


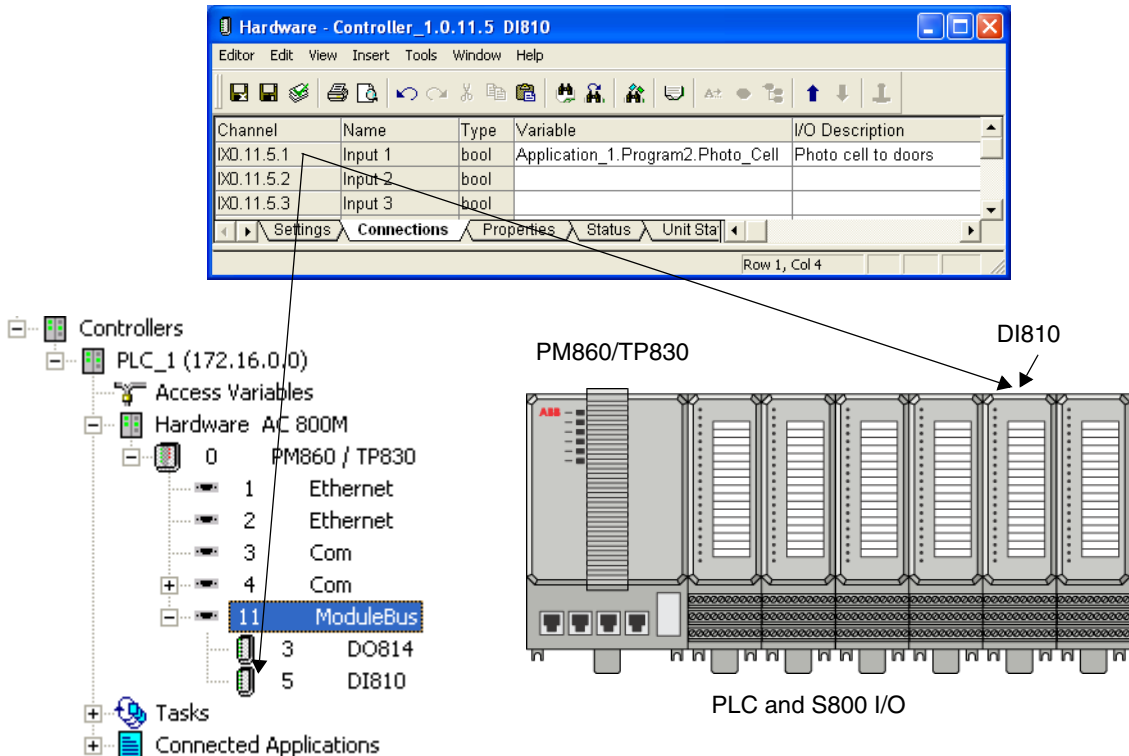
Figure 44. Hardware setup (Project Explorer).



Information on an I/O unit, for example DO814 or DI810, can be accessible from the On-line Help; select the I/O unit in the Project Explorer and press F1.

## Connecting IO to an Application

The hardware address of a hardware unit is composed of the hardware tree position numbers of the unit and its parent units, from left to right, separated by dots. For example in [Figure 45](#), the PM860/TP830 unit at position 0; ModuleBus at position 11; the I/O unit DI810 at position 5, hence the address 0.11.5.



*Figure 45. The relationship between the DI810s physical location and the hardware address in the editor.*

Generally, you add a hardware unit with the “New unit” command in the Project Explorer, and configure the unit with the hardware editor.



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. Note! 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 78, 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 a Variable to DI810 Channel

From 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 **Program2**, 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.

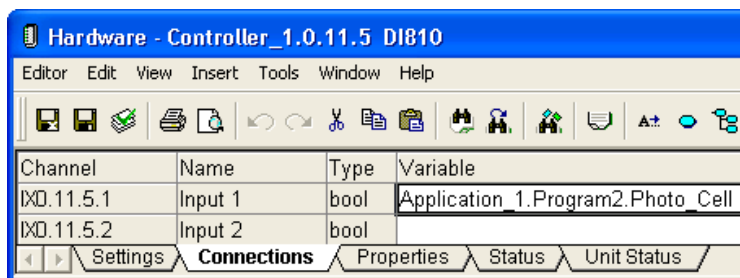


Figure 46. The variable Photo\_Cell connected to first IO Channel.


## Method 2 - Using a Path Selector

### Connecting a Variable to DO814 Channel

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 QXD.11.3.1 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 simply clicking the  icon located in the hardware editor.

4. Expand **Application\_1 > Program2**. Double-click **Motor\_1** to insert the full path, see [Figure 47](#).

Channel	Name	Type	Variable
QXD.11.3.1	Output 1	bool	
QXD.11.3.2	Output 2	bool	
QXD.11.3.3	Output 3	bool	
QXD.11.3.4	Output 4	bool	
QXD.11.3.5	Output 5	bool	
QXD.11.3.6	Output 6	bool	
QXD.11.3.7	Output 7	bool	
QXD.11.3.8	Output 8	bool	
QXD.11.3.9	Output 9	bool	
QXD.11.3.10	Output 10	bool	
QXD.11.3.11	Output 11	bool	
QXD.11.3.12	Output 12	bool	
QXD.11.3.13	Output 13	bool	
QXD.11.3.14	Output 14	bool	
QXD.11.3.15	Output 15	bool	
QXD.11.3.16	Output 16	bool	
QWD.11.3.17	All Outputs	dword	
IWD.11.3.18	Channel status	dword	
IWD.11.3.19	UnitStatus	dint	
			bool

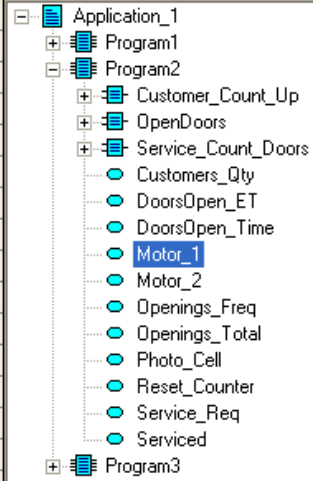



Figure 47. The path for Motor\_1 variable selected in the tree.

5. Place the cursor in the second empty white cell (Channel QXD.11.3.2 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 48](#).

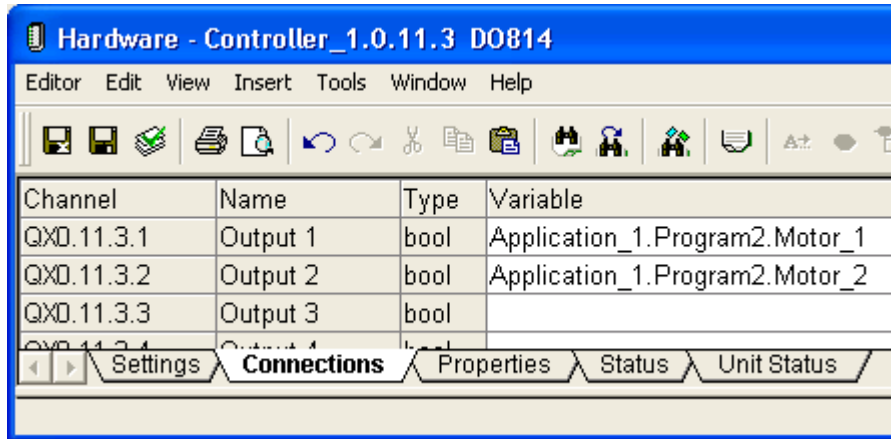




Figure 48. The Motor\_1 and Motor\_2 connected to DO814.

6. Click **Check**  to check the declaration.
7. Click **Save and Close** .
8. Done!

## Reading I/O addresses from the Application

An easy way to read the I/O address is to open (in this case) **Program2** 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 49](#)).

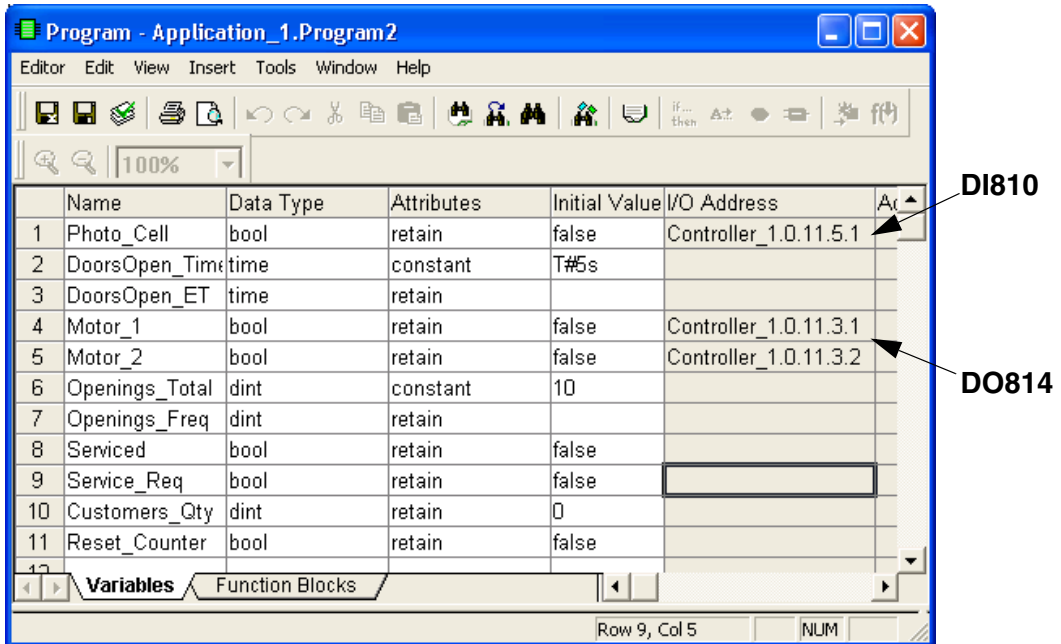


Figure 49. The I/O Address column shows how variables are connected to I/O channels.

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. You are advised to proceed to [Section 6, Connecting the PLC and Go Online](#).



---

## Section 6 Connecting the PLC and Go Online

This section starts with the prerequisites for connecting a PLC. It comprises of downloading PLC firmware and setting a PLC IP address. The last part of this section covers general procedures for downloading a project to the PLC.

If you have created a project according to [Section 4, MyDoors Project](#) and then followed the instructions in [Section 5, Hardware Configuration](#), you will in this section be able to complete the MyDoors project by downloading the application to a PLC.

If you do not have access to a PLC or IO modules, you can still follow this example with a **SoftController**. If you run with a SoftController - you can jump directly to the subsection [Setting the System Identity in Control Builder](#) on page 89.

### Firmware Upgrade

The PLC firmware version and the PLC control builder version must be identical. If you are unsure, perform the steps in this section. It will only take a few minutes of your time.



Firmware upgrade can be performed from the PLC control builder via the Ethernet network (see the Control Builder Online Help) The Serial line upgrade procedure is also available.

#### Firmware Upgrade via the Serial Cable (TK212A)

1. Connect the serial cable between the Control Builder PC and the PLC, as specified in [Table 3](#). For the type of cable, see [Appendix D, Communication Cables](#).

Table 3. Cable connection for the PLC Controller.

PLC	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 PLC.
3. From the Windows Start menu select **All Programs > ABB Industrial IT PLC > AC 800M > Utilities > Serial Firmware Upgrade**. The following dialog box will appear.

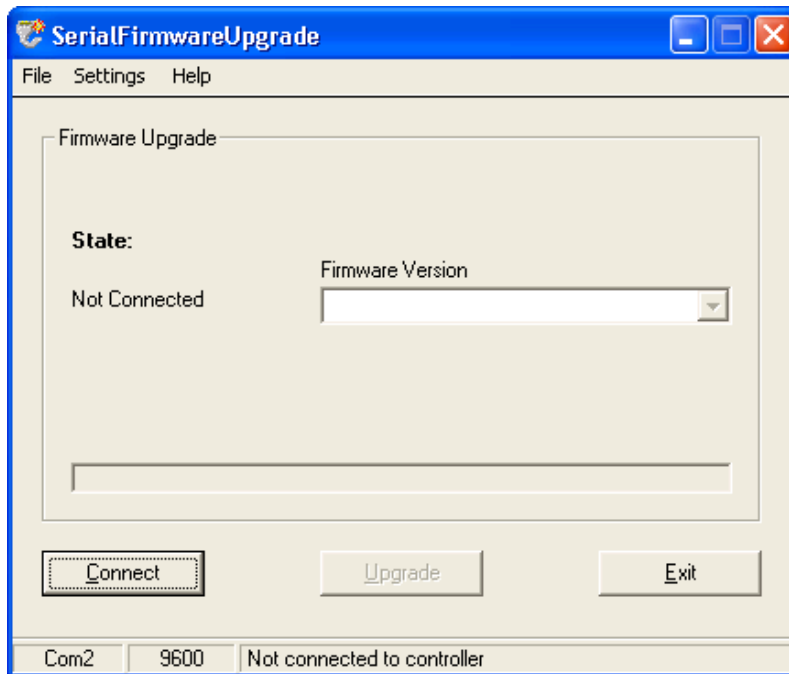


Figure 50. The Serial Firmware Upgrade dialog box.

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

- Click the **Connect** button and press the **Init** push-button on the PLC until the **Run 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 51).

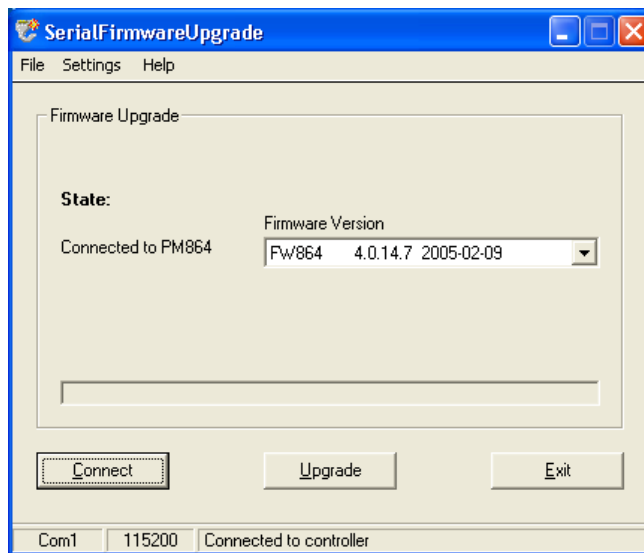


Figure 51. A Firmware version displayed in the text field.



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

- Select Firmware version<sup>1</sup> from the drop-down menu and click **Upgrade**. File transmission starts to the controller. A confirmation window opens when the controller is upgraded, see Figure 52.

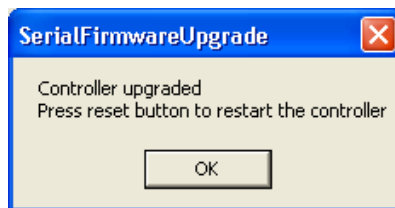


Figure 52. The Serial Firmware Upgrade window.

1. The firmware version must be identical to the installed Control Builder version.

7. Click **OK**.
8. Click **Exit**.
9. Press the **Init** push-button on the PLC until the **Run** LED starts to blink.

## 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 PLC 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, thus any changes made by Microsoft Corporation in Windows must be consulted in your Windows User Documentation. A configuring tool, named IPConfig, is used to set the IP address for the PLC.

### Setting IP Address for PLC

#### Preparations

Connecting the cable between the Control Builder and the PLC are exactly the same as described in [Firmware Upgrade](#) on page 81.

1. Connect a serial cable between the Control Builder PC and the PLC, as specified in [Table 4](#). For the type of cable, see [Appendix D, Communication Cables](#).

*Table 4. Cable connection for the PLC Controller.*

PLC	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 PLC.

### Starting the IPConfig Tool

- From the Windows Start menu select **All Programs > ABB Industrial IT PLC > AC 800M > Utilities > IPConfig**. An IP Config dialog opens.

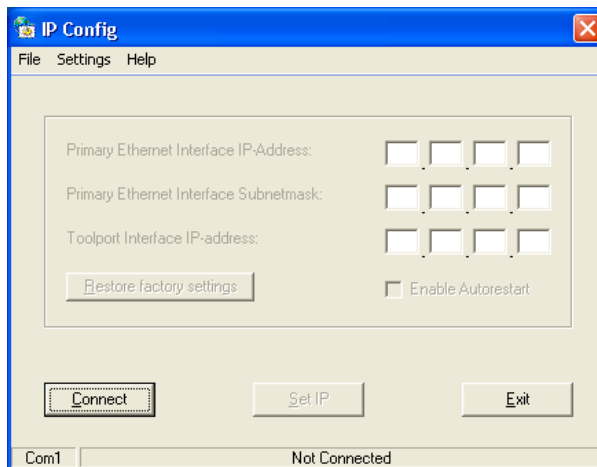


Figure 53. The IP Config dialog box.

- Click the **Connect** button and press the **Init** push-button on the PLC until the **Run LED** starts to blink. Wait about a minute until a message appears, see Figure 54.

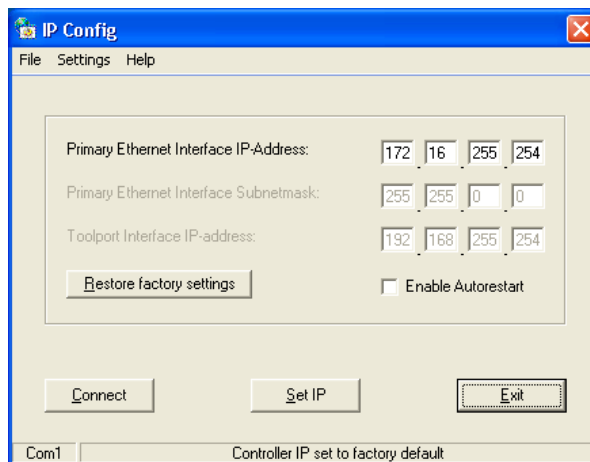


Figure 54. The IPConfig dialog box with factory default setting.



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 55](#).

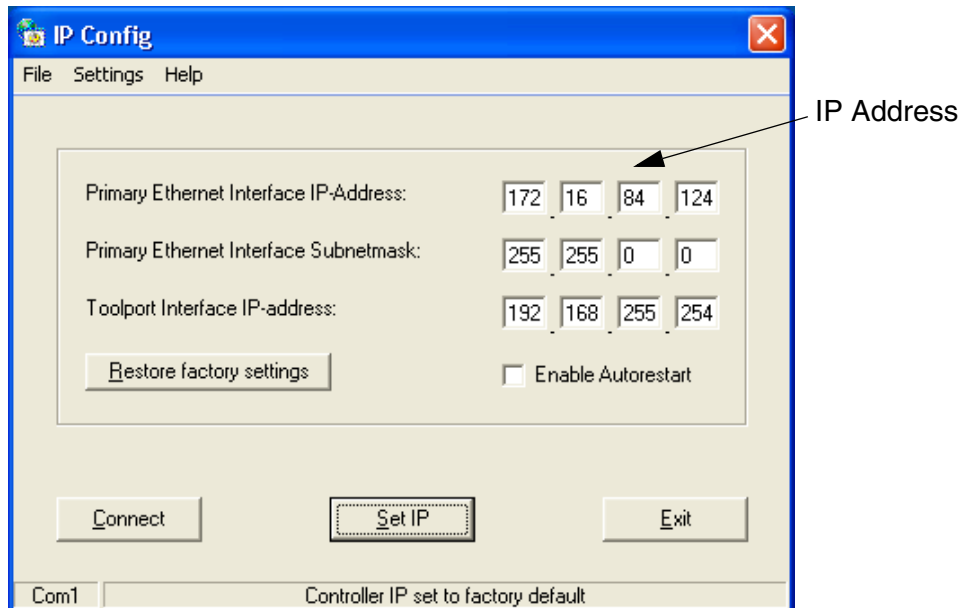


Figure 55. IP Config window for setting unique IP address.

7. Click **Set IP**. The new address will be sent to the PLC and an IP Config window opens, see [Figure 56](#).

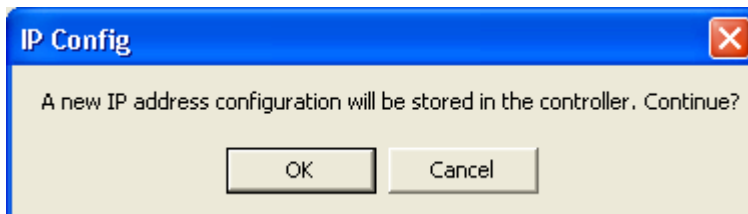


Figure 56. IP Config confirmation dialog window.

8. Click **OK**.

9. Turn off the PLC power switch and then turn on the PLC power switch again. The new IP address is not valid until the controller has been restarted.
10. Press the **Init** push-button on the controller until the **Run** LED starts to blink.

## Setting IP Address for PC

The following instructions will help you setup the IP address (in Windows) for the Control Builder PC.

1. Select **Start > Control Panel**.
2. From Control Panel in Category View, select **Network and Internet connections**.
3. Select **Network Connections**.
4. Right-click **Local Area Connection** and select **Properties**. A Local Area Connection Properties dialog opens.
5. In the 'Connect using' list, select the Ethernet board.
6. Select **Internet Protocol (TCP/IP)** and click **Properties**. An 'Internet Protocol (TCP/IP) Properties' dialog opens.
7. 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 PLC 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 PLC, thus **not** 124 in this example.

8. Enter an IP address, in this example (172.16.84.120) and then enter sub net mask (255.255.252.0).
9. Click **OK**. To close all dialog windows.
10. Connect a network cable. The port and channel positions are shown in [Table 5](#).



To check that the IP configuration works; open the command prompt DOS window and ping the PLC by writing the following command: ping 172.16.84.n, where "n" is the address selected for the PLC.



If the PLC 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 PLC and the PC, then use a *cross-over* Ethernet cable.

*Table 5. Channel positions for connecting the Ethernet cable in the PLC.*

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



CN2 port on the PLC must not be connected to the network. This port is used for connecting the PLC 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 PLC and Go Online!

### 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 PLC\_1

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

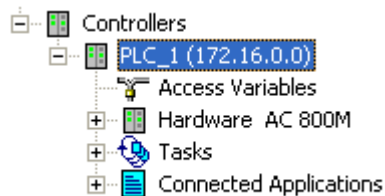


Figure 57. The Controllers expanded in Project Explorer.

2. Right-click **PLC\_1** and select **System Identity**. The System Identity window opens.

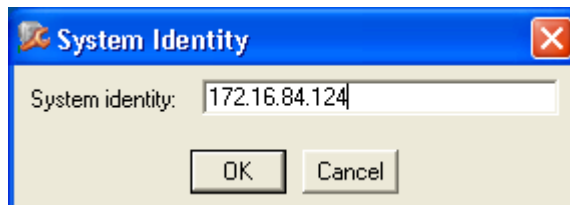


Figure 58. The System Identity window for setting the IP address.

3. Enter the IP address of the controller (for example: 172.16.84.124) and click **OK**. The System Identity window closes.

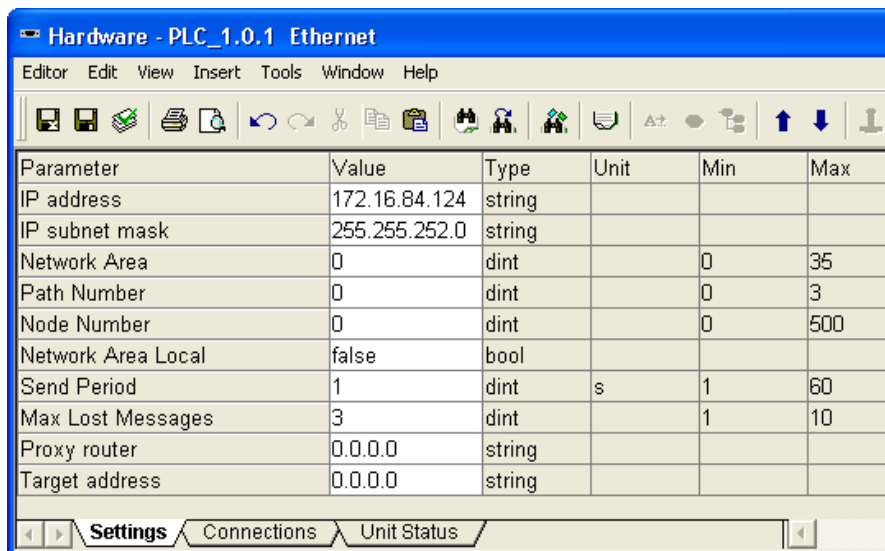


If you run with a **SoftController** then type in your computer IP address and finish with colon and the digit 2. Example: 10.46.35.117:2

4. Expand **Hardware AC 800M** until you find **1 Ethernet**. Right-click the **Ethernet** icon (at position 1) and select **Editor**. The editor opens.
5. Select the **Settings** tab (lower left corner, see [Figure 59](#)) and enter the IP address in the IP address Value field.



Note that the IP address of the first Ethernet port has to be the same as the IP address of the PLC (system identity). The second Ethernet port (at position 2) is only used if the controller is connected to a redundant network. For more information about redundant networks, study the subsection [Setting Up Redundant Network](#) on page 117



*Figure 59. IP address for controller Ethernet port at position 1 (in this case 172.16.84.124 which is the same IP address as given in System Identity).*

Click **Save and Close**

## Downloading the Project to the PLC

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


If you run with a SoftController go to [Downloading to the SoftController](#) on page 92.



You must make sure that the PLC and all other hardware units have the right firmware. For instructions on how to check and upgrade firmware versions, see [Firmware Upgrade](#) on page 81.

### Downloading to the PLC

The following instructions address the project MyDoors, which was previously created in [Section 4, 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 **Cold Restart All**.
4. Click **Continue**.

- The Difference Report function is enabled by default, thus a window will open unless disabled.

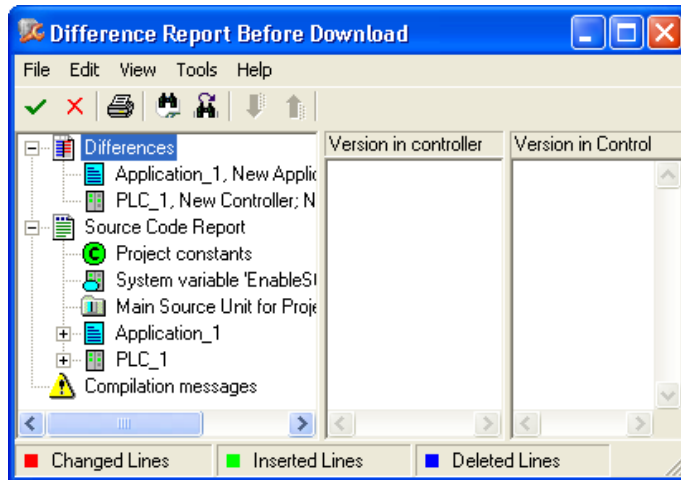


Figure 60. Difference Report Window.

5. Click the green check mark (see Figure 60) to continue.

### Downloading to the SoftController

Make sure that your (MyDoors) project is in Offline mode (not running in test mode). From Project Explorer, expand the Controllers folder:

1. Right-click PLC\_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:

2. **Start > All Programs > ABB Industrial IT PLC > AC 800M > SoftController > SoftController.** The SoftController start panel opens.
3. Click the **Start** button. The Status field displays *Started* and the SoftController starts.

From Project Explorer:

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

5. Click **Cold Restart All**.
6. Click **Continue**. You should now be Online!

## Test the Program Online

This subsection describes how to force 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 **Program2** and select **Online Editor** to open the online editor.
2. Right-click I/O module **DI810** and select **Editor**.
3. Check 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 for program **Program2** to start and stop the motors. Now, you are using the I/Os to control the motors (See [Figure 61](#) and [Figure 62](#)).

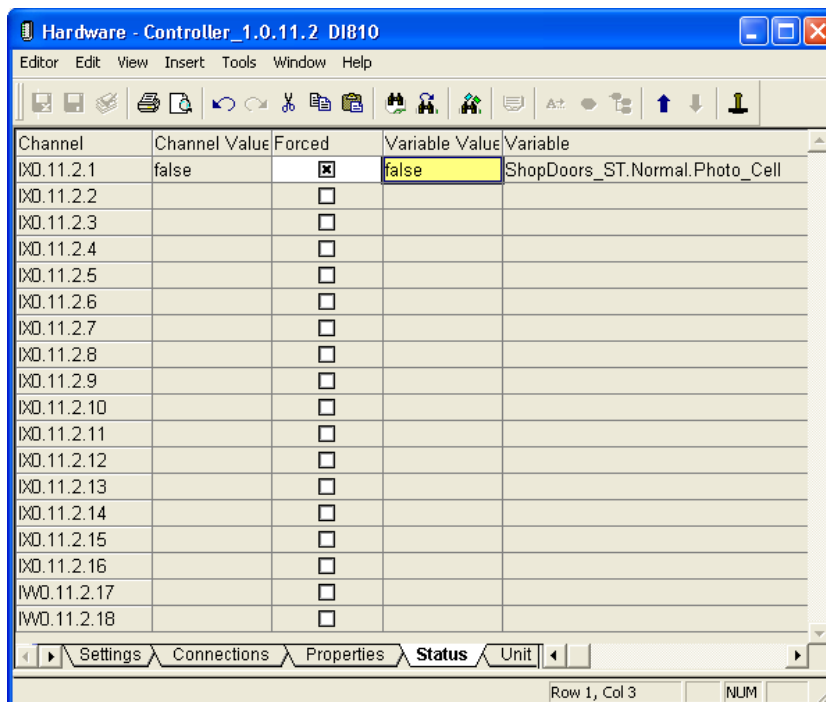


Figure 61. The status of the photocell and the motors can be forced in the I/O editor.

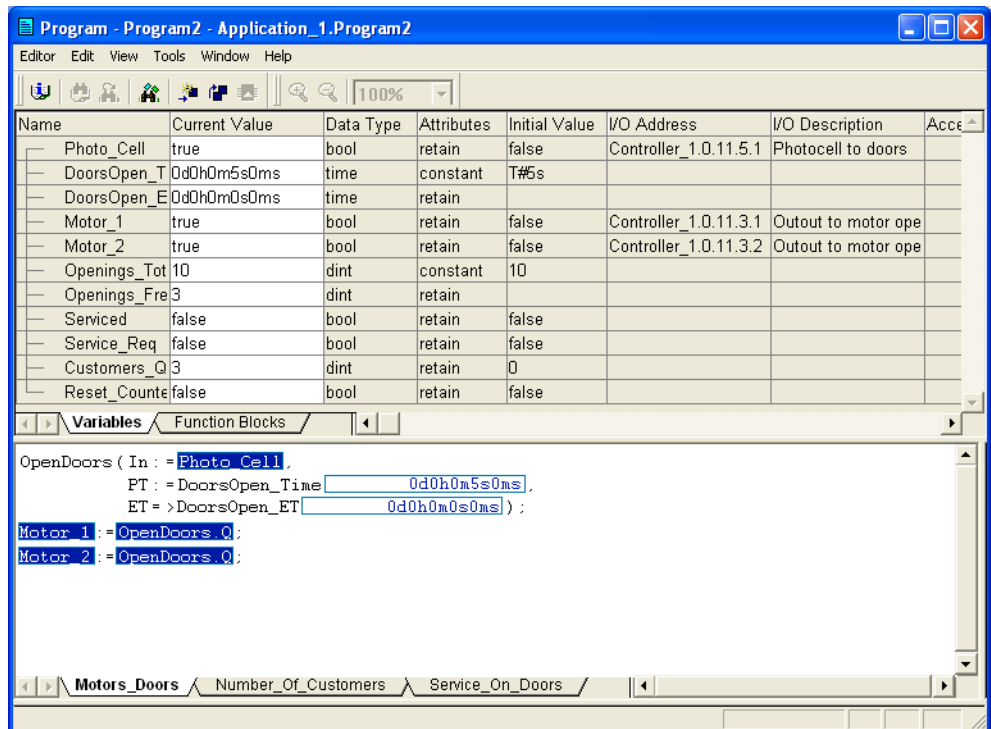


Figure 62. The changing motor values can be inspected in the online editor.

4. Close all editors.





---

# Appendix A PLC Control Builder AC 800M Settings

This section starts with the Setup Wizard for PLC Control Builder. If you are setting up a single user configuration with minor adjustments, the wizard will guide you through the settings. However, if you are going to setup a multi-user configuration you will have to change the file path in the last dialog, 'File Location' under Product settings.

Having that said, if this is your first time running the Wizard, you can always change the default path location later on after reading the subsection [Multi-User Configuration](#) on page 104.

At the end of this section you will learn how to enable/disable Difference report dialogs before testing, and downloading applications to PLC.



If you are content with the default settings for a single-user configuration, you do not have to complete the two Setup Wizards in this appendix for working with PLC Control Builder and downloading to a PLC.

## Starting the Setup Wizard for PLC Control Builder

From **Start** select **All Programs > ABB Industrial IT PLC > AC 800M > Utilities > Setup Wizard**.



Some tabs have an Apply button. Specified settings will not be implemented until this button is clicked (see [Figure 63](#)).

All *Setup Wizard* dialog boxes contain a Show Settings button. Click this button to open a log file on screen containing all available Wizard settings. It also contains a list of system environment variables.

## System Settings for PLC Control Builder

### Auto Logon to Windows

With this function enabled, the Windows operating system will automatically logon a selected user when the PC is started. This function should only be enabled if Control Builder auto startup is needed.

Select **Enable Auto Logon** and Enter a user name (set in Windows), password (twice) and the computer name (see [Figure 63](#)). If an unknown user name is entered, the Auto-logon function will fail.

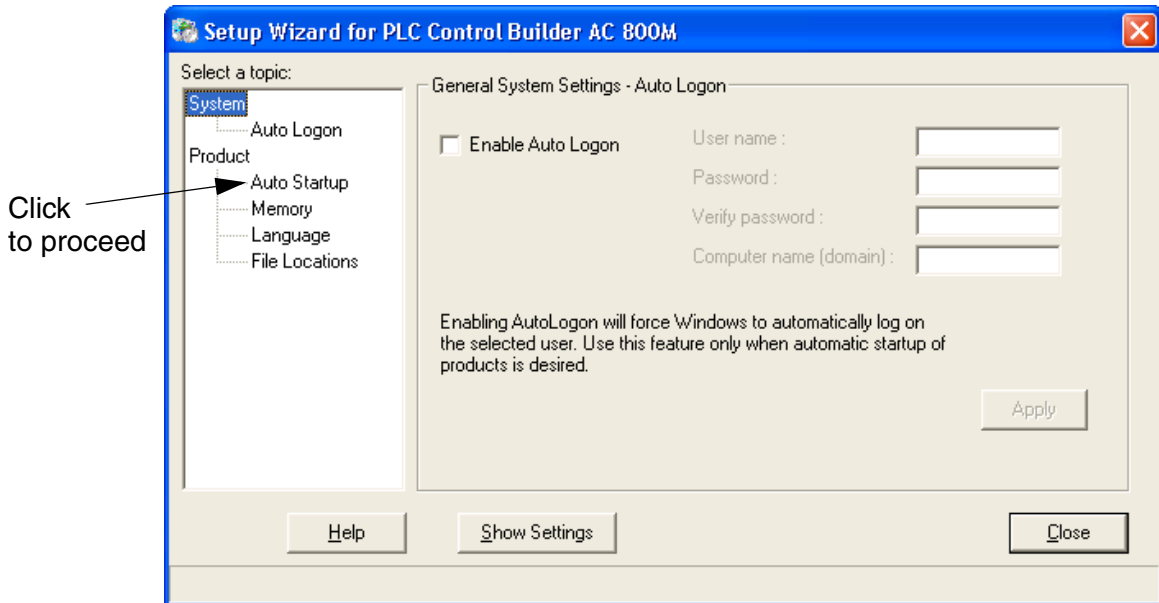


Figure 63. Setup Wizard start menu for PLC Control Builder. Click the topic 'Auto Startup' to proceed to next step in the wizard.




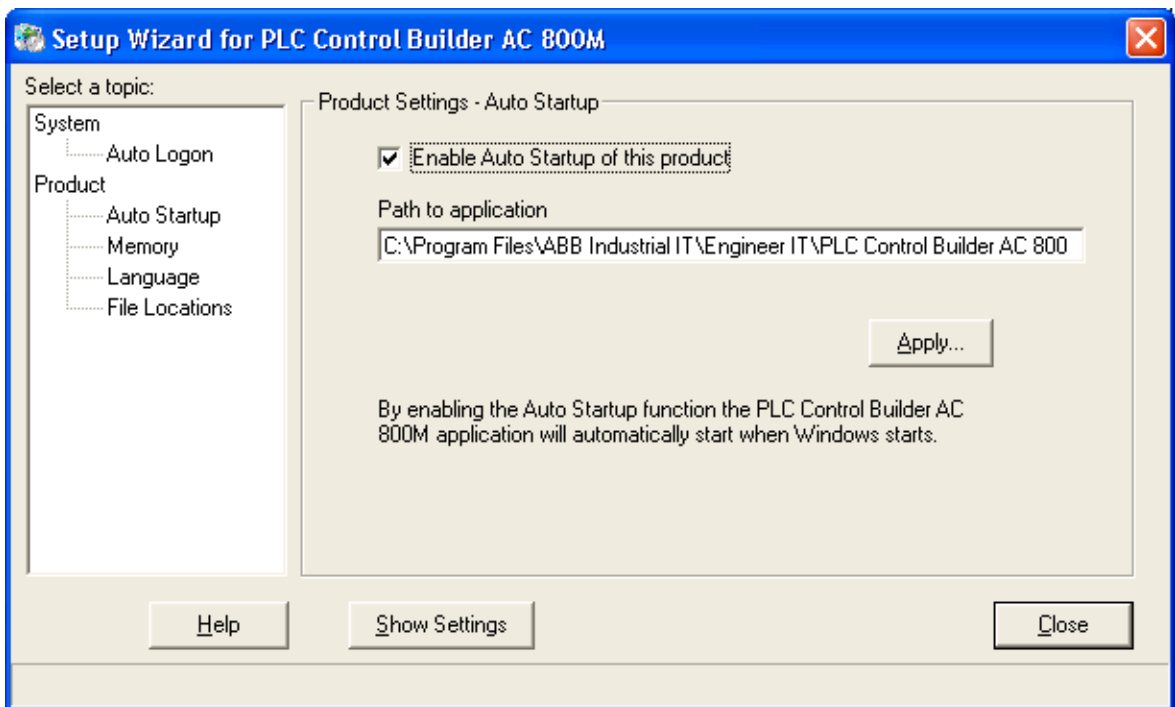
If you set the Enable Auto-Logon function, it will remain enabled even if you remove PLC Control Builder.

## Product Settings for PLC Control Builder

### Auto Startup of Control Builder

Auto Startup will start the Control Builder automatically, as soon as the user logs in to Windows. However, the Auto Startup function requires that the Auto Logon function under System has already been enabled.

 PLC Control Builder will start without any application being opened. The path to Control Builder should be entered in the dialog box (see [Figure 64](#)).



*Figure 64. Product settings for Auto-Startup. Click the topic 'Memory' to proceed to the next step in the wizard.*

## Memory Reservation

The total memory reserved for an application is based on a calculation involving the size of the physical RAM memory and the hard disk paging file. The following is shown in the Memory topic (see [Figure 65](#)).

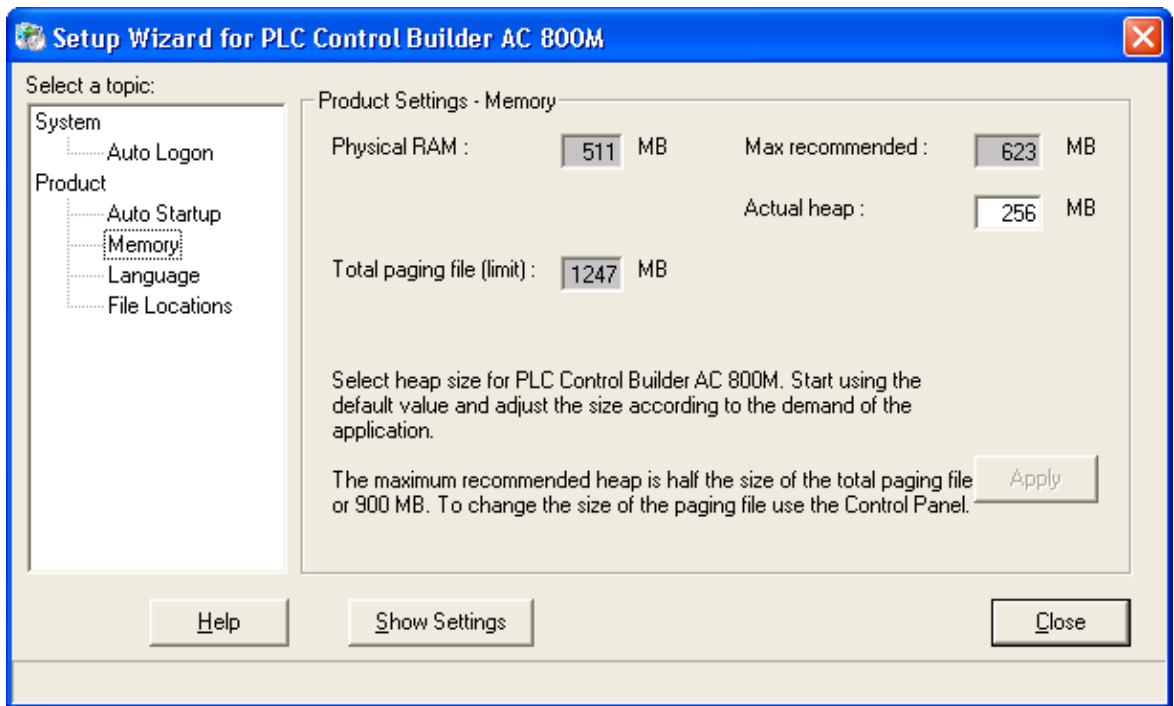
- Physical RAM memory
- Total paging file (this file can be changed in the Windows control panel)
- Maximum recommended heap
- Actual heap (should normally be increased)

The actual heap should be set to the size of the application but not exceed the Maximum recommended heap size, see memory settings in [Figure 65](#).

The default PLC Control Builder heap size is 256 MB. However, in conjunction with large projects, this has proven not to be sufficient, and as a result the PLC Control Builder has crashed. If this should occur (the server cannot start), increase the heap size by simply doubling the previously allocated heap size.



The amount of free memory can be checked by opening the About... dialog box in the Control Builder Help menu. Heap size should be increased when less than 30% remains.



*Figure 65. Setting the memory Heap size dialog. Click the topic 'Language' to proceed to next step in the wizard.*



Do not forget to click Apply to activate the changes.

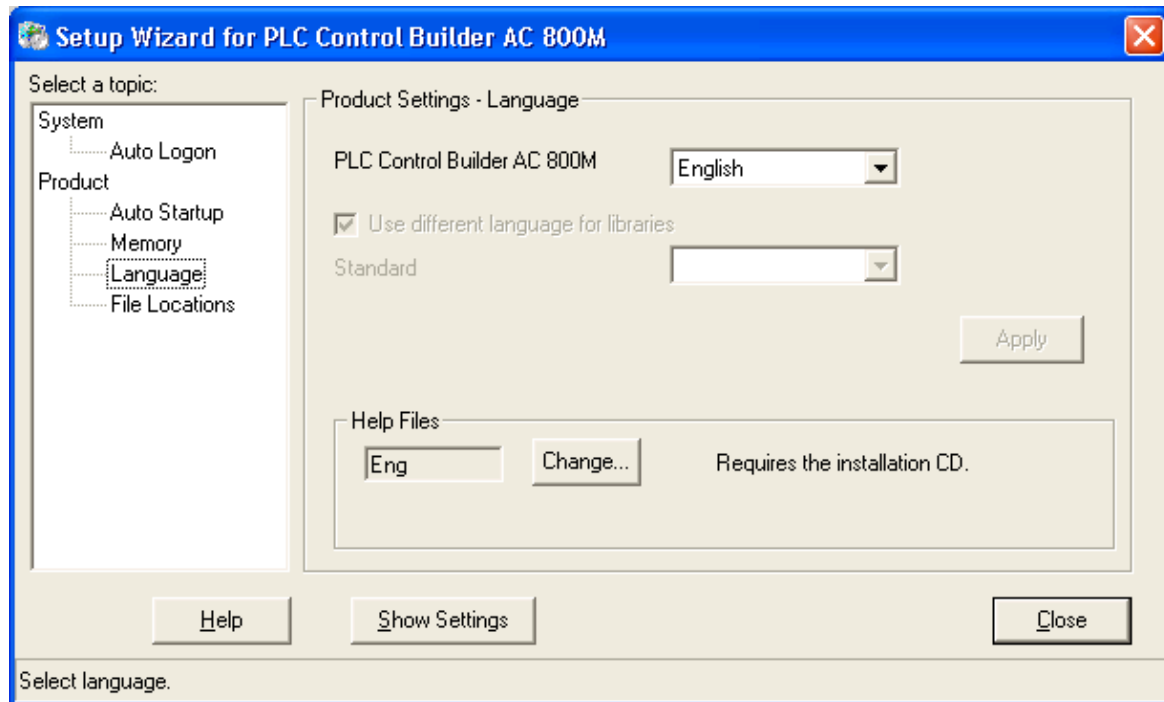


A heap size must not be larger than the paging file. When the heap size is saved, the system checks this.

If the heap size is larger than the maximum recommended (half the total paging file), a warning message will advise you to change the size. If the heap size is larger than the Total paging file, an error message will tell you to reduce the size.

## Language

The language to be used for the programming tool, the libraries, and the Online Help files is selected under the Language tab (see [Figure 66](#)). The installation program will, by default, select the language set in Windows XP or Windows 2000. If the product does not support the language, English will be used as the default language.



*Figure 66. The Product setting dialog for language. Click the topic 'File Locations' to proceed to next step in the wizard.*

## File Locations

PLC Control Builder has three file locations which can be managed by the Setup Wizard. The Working folder and the MMS server working folder are handled by the system, thus should never be modified. The Project folder should only be changed, if the PLC Control Builder station should be part of a multi-user configuration.

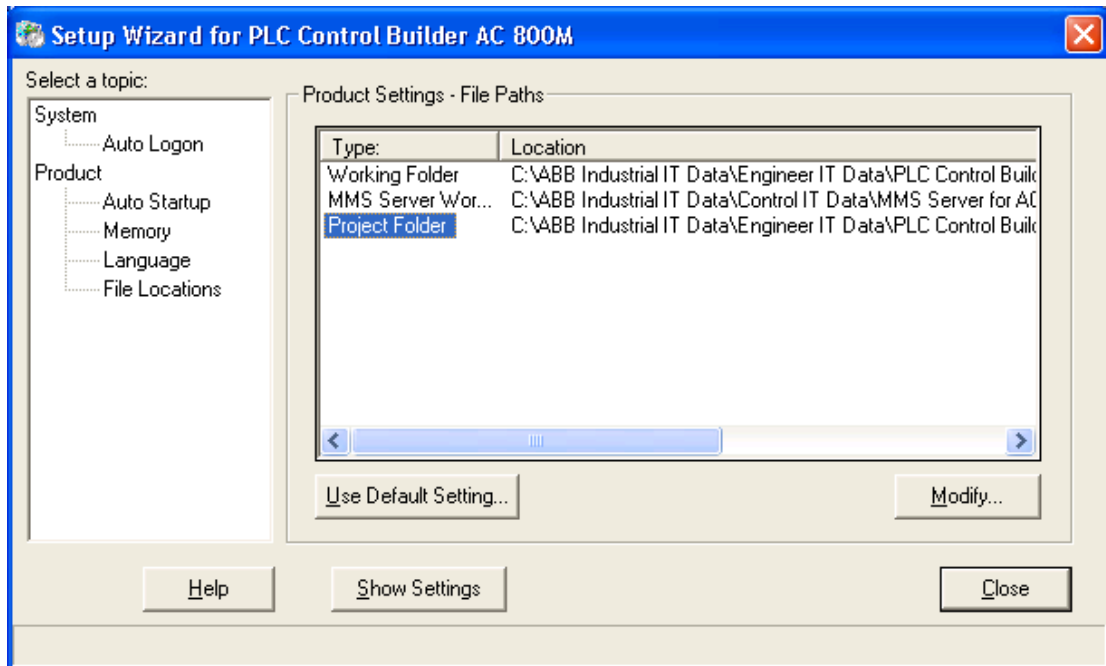


Figure 67. Product topic for file locations.

- Working folder; contains PLC Control Builder station log files, workspace settings etc.
- MMS Server working folder; contains MMS Server log files.
- Project folder; contains projects.

Clicking *Use Default Settings*, will reset your personal settings to the general default settings.



How to change the file location for the Project Folder, see [Multi-User Configuration](#) on page 104.

## Multi-User Configuration

You must install PLC Control Builder on every PC machine, before setting up a multi-user environment. You can also install an OPC Server on one of the PLC Control Builder stations or on a standalone PC machine. Furthermore, all PC machines must be connected to the same network.

A multi-user configuration provides that all PLC Control Builder stations and an OPC Server have access to the common project files<sup>1</sup>. During multi-user engineering all Control Builder stations must write/read engineering changes to the common project files located in a shared project folder. The OPC Server reads runtime data directly from the PLC over the network, but in order to do so it must have access to the configuration data located in the common project folder. Thus, the project folder must be both shared and placed on a network server, before your team can start multi-user engineering.

Multi-user configuration consists of the following sections:

- [Creating a Shared Project Folder](#) on page 105.
- [Setting Up PLC Control Builder Stations](#) on page 107.
- [Setting Up OPC Server](#) on page 109.
- [Configuration Example](#) on page 114.
- [Guidelines for Multi-User Engineering](#) on page 115.

---

1. A PLC Control Builder project contain several files. These project files hold configuration data for libraries, applications, hardware, project constants etc.



## Creating a Shared Project Folder

Select a PC station as the network File server (from now on MyServer). Start Windows and log in with administrator role.

1. Create a project folder (from now on **AC800Mprojects**) on MyServer.
2. From **Start** select **All Programs > Accessories > Windows Explorer**. The Windows Explorer opens in MyServer.
3. Right-click **AC800Mprojects** and select **Sharing and Security** from the context menu. A Properties dialog opens.
4. Select **Sharing** tab and click **Select this folder** radio button. Locate the settings from the Properties dialog in [Figure 68](#).

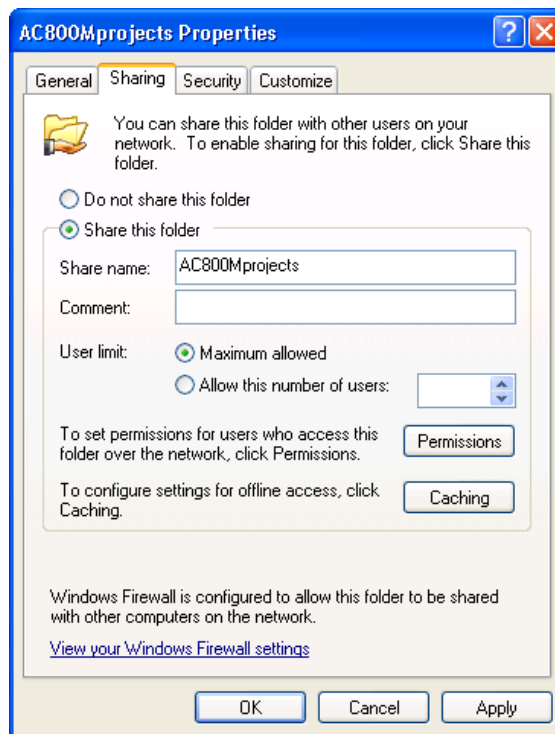


Figure 68. Property dialog with the Sharing tab active.

5. Click the **Permission** button to open the permission dialog ([Figure 69](#)).

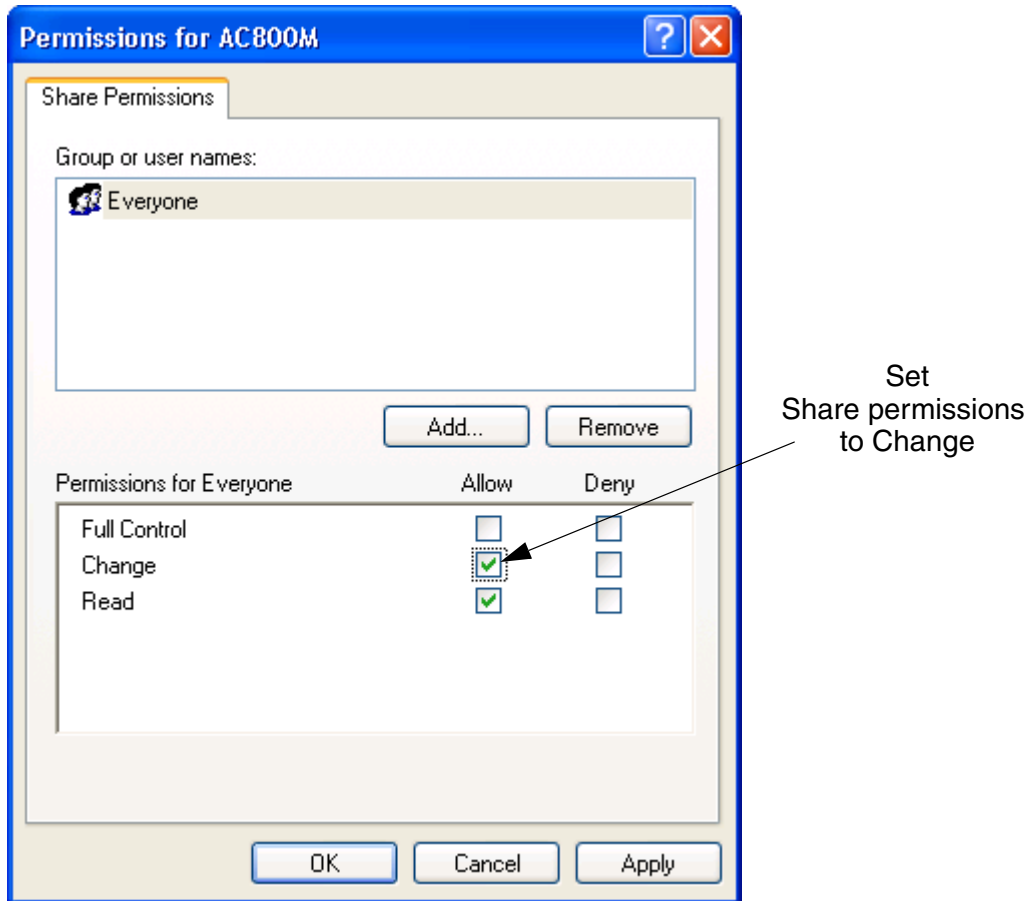


Figure 69. Share permission dialog for the shared project folder AC800Mproject.

6. Select **Change** and click **Apply**.



It is preferable to create a new user group (for example MyTeam) in Windows and add your project members to MyTeam group. You can then set the permission Change for MyTeam instead of the Everyone group which is a general group and ought not to have permission Change.

7. Click **OK**. Permissions dialog closes.
8. Click **OK**. Properties dialog closes.

## Setting Up PLC Control Builder Stations

Make sure PLC Control Builder has been installed and the PC station is connected to Ethernet. For installation instructions see [Section 2, Installing PLC Software](#).

1. Start the **Setup Wizard** according to [Starting the Setup Wizard for PLC Control Builder](#) on page 97.
2. Select **File Locations > Project Folder** and click the **Modify** button. A browse dialog will open and assist you in locating the Project folder.

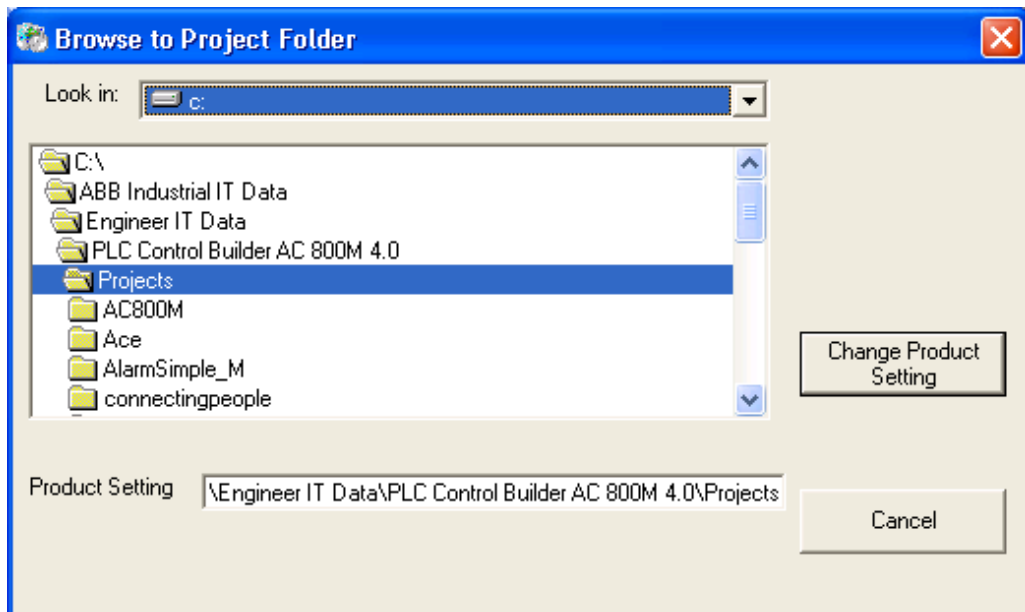


Figure 70. A Browse dialog for locating the shared project folder



The path must be specified with an UNC path that contains the server name and the shared folder name: \\MyServer\AC800Mprojects

3. Locate the network File server and browse to the shared **AC800Mprojects** folder.
4. Click the **Change Product Setting** button. The Browse dialog will close.

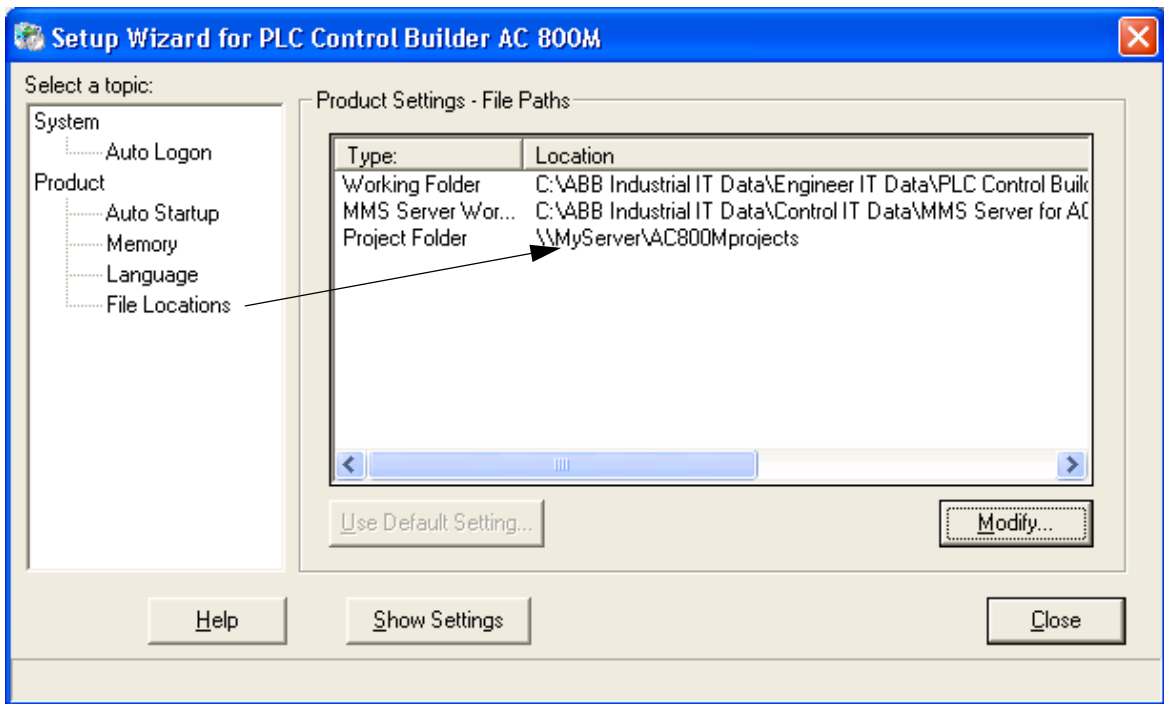


Figure 71. The new file location for the common project folder.



For more information about the differences between the Working folder and Project folder, see [File Locations](#) on page 103.

5. Click the **Close** button. Setup Wizard will close.
6. Repeat these steps for all PLC Control Builder stations.

## Setting Up OPC Server

An OPC Server configuration for multi-user engineering requires basically two things; first you need to select the path to the common project folder under the topic File Locations in the Setup Wizard and secondly you must set up a Service Account.

Make sure the OPC Server has been installed. For installation instructions see [Installing the OPC Server for AC 800M](#) on page 31.

1. From **Start** select **All Programs > ABB Industrial IT PLC > AC 800M > OPC Server for AC 800M > Setup Wizard**.

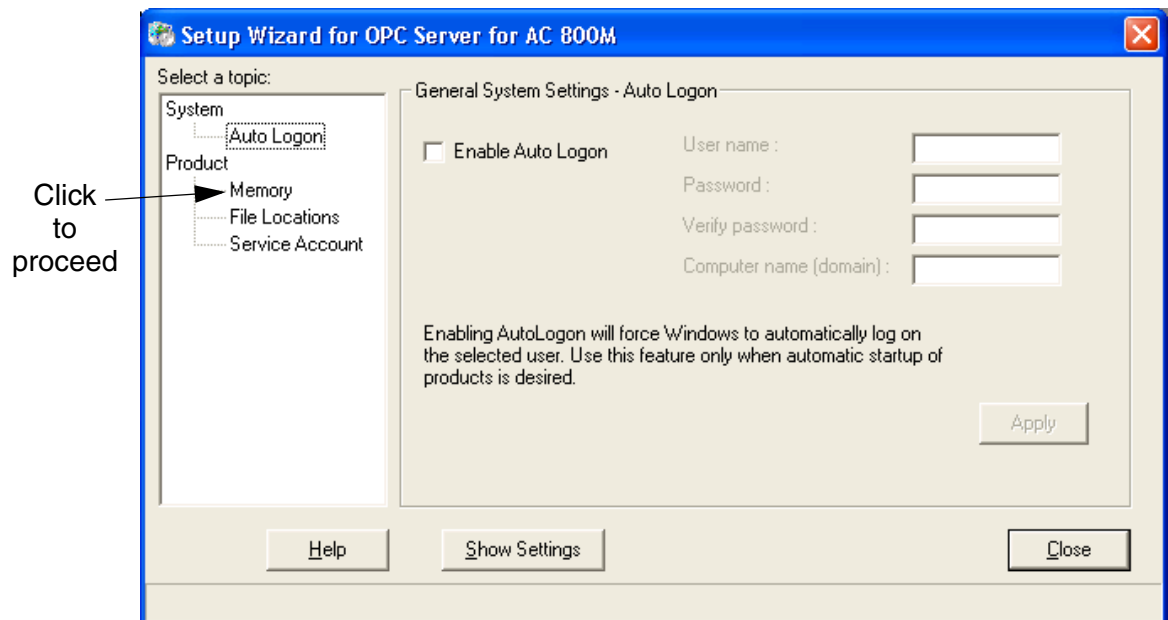
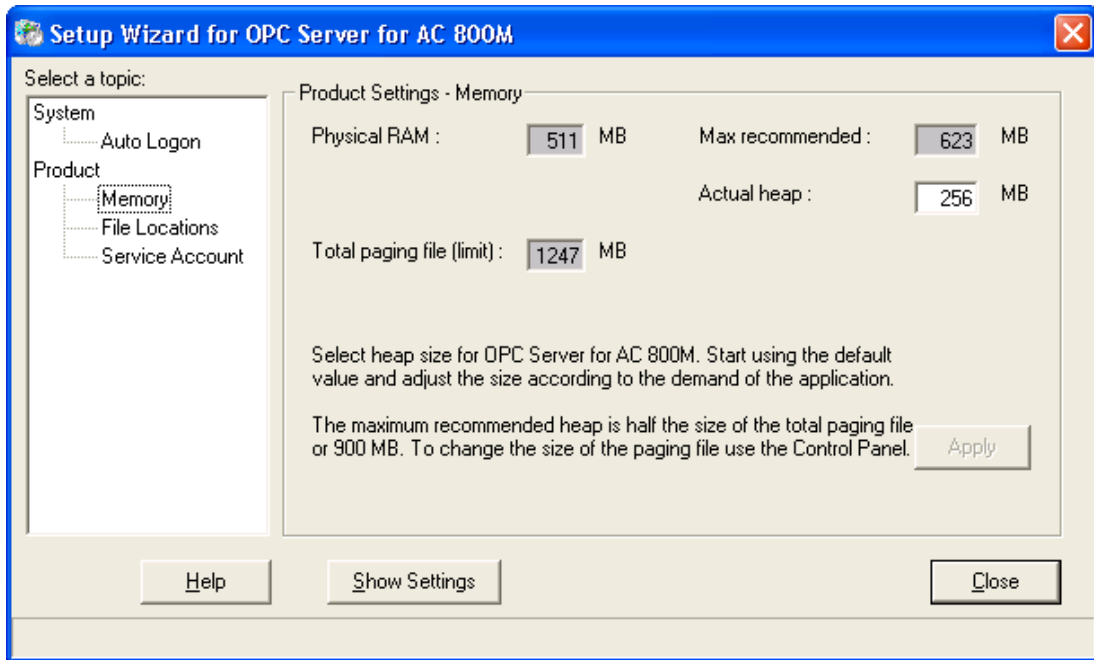


Figure 72. Setup Wizard start menu for OPC Server.

2. Select **Memory**. A memory setting dialog will open.



*Figure 73. Memory settings for OPC Server. After you are done with the memory settings, click the topic 'File Locations' to proceed.*

The actual heap should be set to the size of the application but not exceed the Maximum recommended heap size. Start by using the default value and then increase the value according to the application demands.

The default OPC Server heap size is 256 MB. However, in conjunction with large projects, this has proven to be not sufficient, and as a result the OPC server has crashed. If this should occur (the server cannot start), increase the heap size by simply doubling the previously allocated heap size. Check the OPC Server's About box shortly after the Server is up running again, and make sure there is at least 30% spare heap.

3. Select **File Locations > Project Folder** and click the **Modify** button. A browse dialog will open and assist you in locating the Project folder.

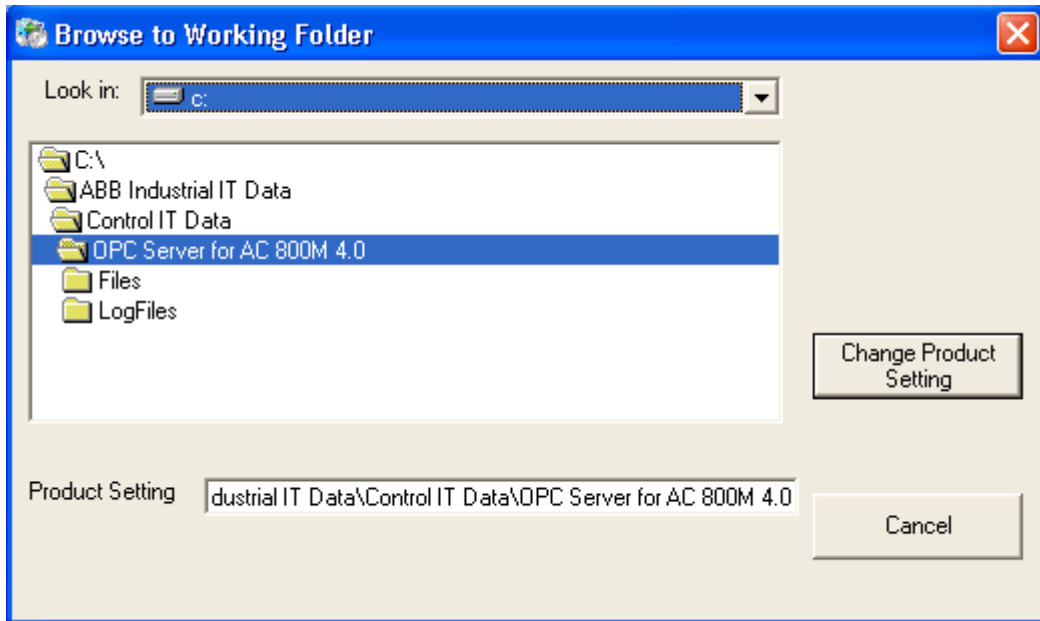


Figure 74. A Browse dialog for locating the shared project folder



You must first map a network drive in Windows, before you can browse to the (UNC) path. Furthermore, the browse dialog does not permit browse navigation on network places. If this is the case, then you must type in the path in the Product Setting text field (Figure 74).

The path must be specified with an UNC path that contains the server name and the shared folder name: `\\MyServer\AC800Mprojects`

4. Locate the network File server and browse to the shared **AC800Mprojects** folder.
5. Click the **Change Product Setting** button. The Browse dialog will close.

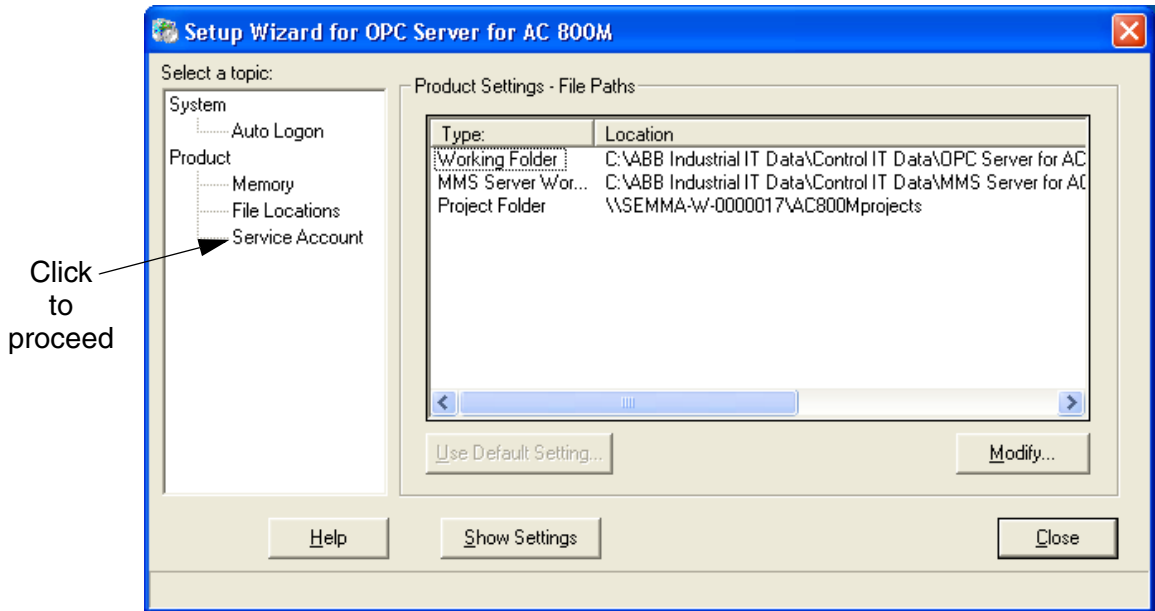


Figure 75. An example of a path to a shared project folder located on a network server. Click the topic ‘Service Account’ to proceed to the next step in the wizard.

6. Select the topic **Service Account**.

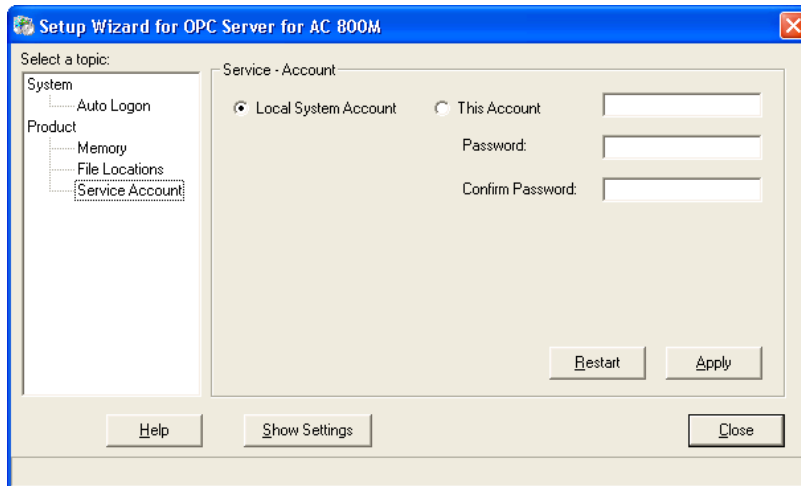


Figure 76. The Service Account Topic for OPC server.



### Which Service Account shall I choose?

The choice between '*Local System Account*' and '*This Account*' depends mostly on the OPC Server location and what else software products are installed on the same PC (as the OPC Server). Here follows a brief presentation of the two account options:

**Local System Account** means basically that (login) Windows User has the privilege permission (Change) to the shared folder (AC800Mproject).

**This Account** means basically that a (login) Windows User has not the privilege permission (Change) to the shared folder (AC800Mproject). A Windows User with that privilege permission must be specified with user name and password.

By selecting *This Account*, any other Windows User (operators, maintenance personnel etc.) can login as Windows user without interrupting the OPC Server traffic.

### Communication Failure between OPC Server and the OPC Panel

If a communication failure between the OPC Server and the OPC Panel occurs, thus a pop-up menu displaying 'Access denied' and the OPC Panel stops responding, proceed with the following steps:

- a. In Windows, search for the **OPCServerPanel.exe** file<sup>1</sup>.
- b. Right-click **OPCServerPanel.exe** and select **Run as** in the context menu.
- c. Setup which user account for running the OPC Server Panel application. Select a user with administrator role.

For more information about setting up user account in Windows, consult your Windows User Documentation.

7. Click the **Close** button. Setup Wizard will close.



Create a small project in one of the PLC Control Builder stations and verify that the project can be opened from all the other PLC Control Builder stations.

---

1. Normally located at C:\Program Files\ABB Industrial IT\Control IT\OPC Server for AC 800M 4.0\Bin\

## Configuration Example

Assume the following network configuration;

- The OPC Server is installed on a PC together with an operator interface. The OPC Server has the Project folder path set to the File Server.
  - OPC Server copies the configuration data from the project folder to its own local working folder. It uses the configuration data to translate the live data traffic from the PLC.
  - OPC Server writes cold retain values to the shared project folder. This is one of the reasons you must set the permission *Change* in Windows.
- If *Local System Account* (see Figure 76) has been selected previously in the Setup Wizard; only members of MyTeam group should login to the PC machine, or the OPC Server will be interrupted.
- If *This Account* (see Figure 76) has been selected previously in the Setup Wizard; any user including an operator can login to the PC machine without interrupting the OPC Server traffic.
- Two PLC Control Builder stations with their project folder shared on the network File Server.
- One File Server with the shared project folder AC800Mprojects.
- A control system here symbolized as PLC.

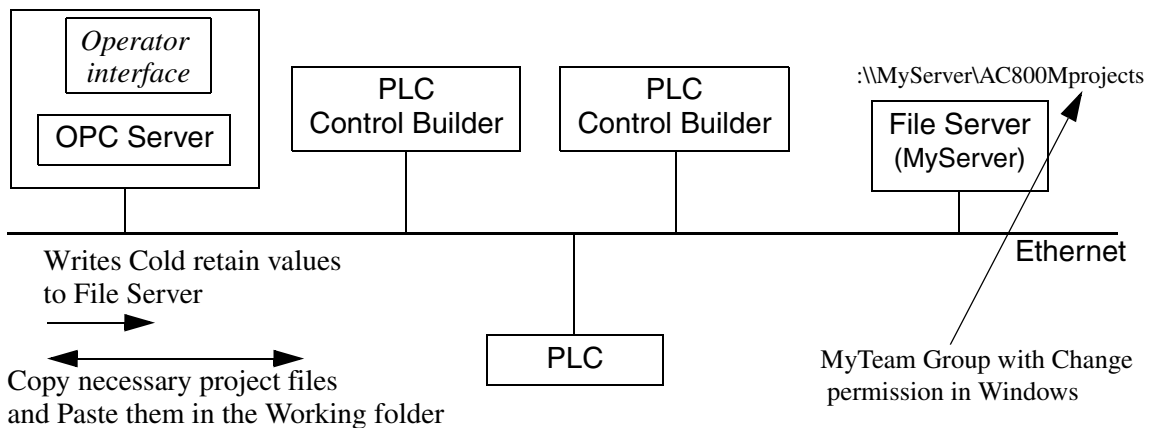


Figure 77. An example of a multi-user configuration.

## Guidelines for Multi-User Engineering

A Project contains a number of files, whereas every member in a multi-user environment has a decisive impact on these project files. This means that a project folder shared by several PLC Control Builders may be subjected to multiple changes at the same time. Therefore, to avoid unwanted read/write results on the project files, you are advised to read these simple guidelines.

1. Several members may work with different libraries etc. without difficulties, but always strive to assign one member to a specific library, application or PLC at the time.
2. If several members must work with the same library or application, then permit only one member to work with a specific Type (Program, Function block type etc.) at the time.
3. Permit only one member to work with control modules in an application at the time.
4. If several members must work with a specific PLC, then permit only one member to work with a specific Hardware unit, Task or Access variables, at the time.



If a rename operation affects several files, Control Builder will first alert you and display the corresponding files before proceeding with the rename operation.

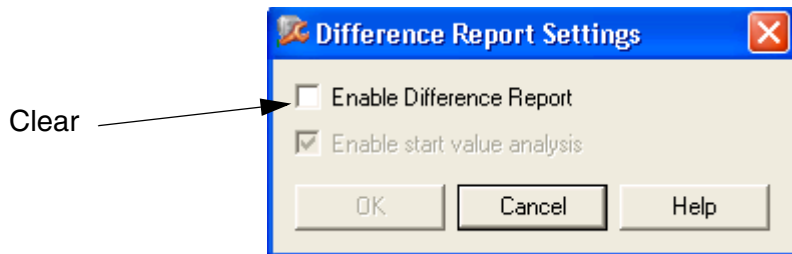
## Disable/Enable Difference Report

The following instructions will disable the Difference Report, thus preventing the Difference Report Settings dialog to popup when selecting *Test Mode*, *Online*, *Download Project* and *Go Online*.

### Disable Difference Report

From the Control Builder Menu bar:

1. Open **Tools > Difference Report Settings**. A 'Difference Report Settings' dialog opens.
2. Clear **Enable Difference Report** option.



3. Click **OK**.

---

# Appendix B Network Redundancy



The information given in this Appendix applies only to users who intend to set up Redundant Networks.

For more information about Redundant Networks and clock synchronization, see online help and the manual *Communication, Protocols and Design*. Study in particular the MMS section.

## Setting Up Redundant Network

The following example teaches you to set up two separate redundant networks with the so called *implicit IP addressing* method. You will learn how to configure IP addresses for two controllers and two PCs. The following sub-sections have step-by-step instructions which you can apply to your own project. After you have completed this example you should be able to add another PLC or PLC Control Builder station.

### Two Separate Redundant Networks

The example consists of PLC\_1 and PLC\_2 on one redundant Control Network, and PC\_1 with a Control Builder on another redundant Client/Server Network. PC\_2, with for example an OPC Server to access the right network components, connects the two separate redundant networks according to [Figure 78](#).

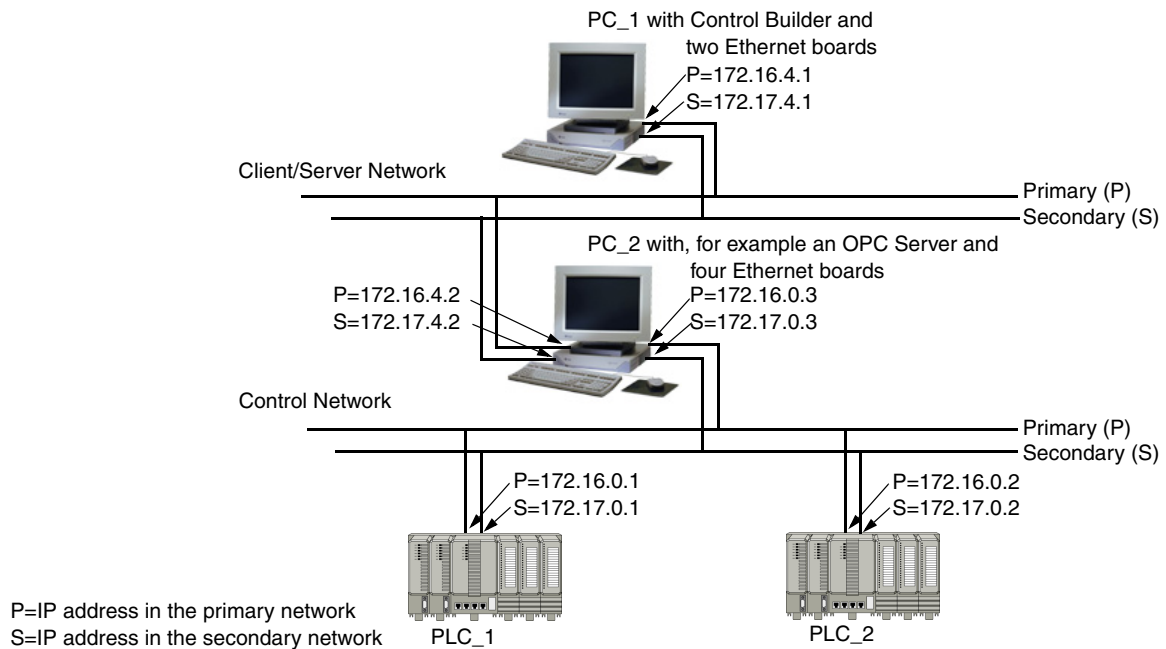


Figure 78. Redundant Control Network and redundant Client/Server Network.

### Changing in RNRP Setup Wizard

You must run the RNRP Setup Wizard for PC\_2, otherwise the routing from PC\_1, via PC\_2, to the PLCs will not work.

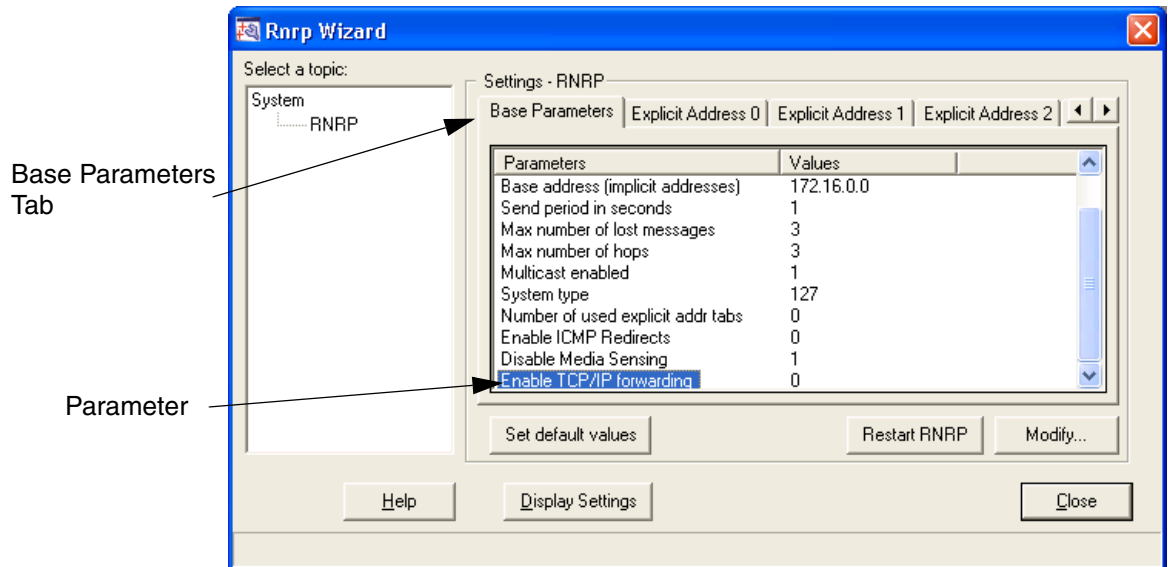
1. Right-click the **ABB RNRP**-icon located at the lower right-side of your Windows desktop. The RNRP Wizard opens. Note you must right-click the icon for opening the Setup Wizard.



Right-click ABB RNRP-icon

Figure 79. ABB RNRP-icon for opening the Setup wizard.

2. Make sure the **Base Parameters** tab are active.



3. Select the **Enable TCP/IP forwarding** parameter and click the **Modify** button. A value dialog opens.

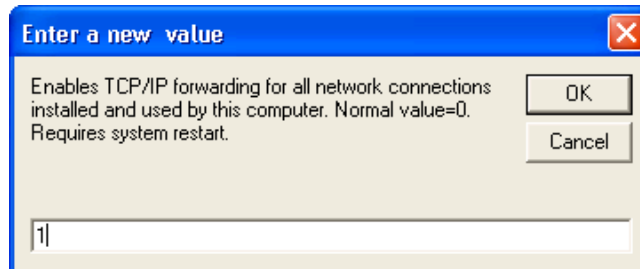


Figure 80. Parameter dialog for changing parameter values.

4. Change the parameter value to **1** instead of the default value 0 and click **OK**.
5. Click **Close**. The RNRP Wizard will close.

## Decide IP Addresses

First you must decide on IP addresses, and the IP sub net mask to use for each Ethernet connection port. In these redundant networks you must have two IP addresses for each PLC and in PC\_1 with the Control Builder. In PC\_2, with the OPC Server, you must have four IP addresses to the four ports in the Ethernet PC boards.

In this example, you select the IP addresses displayed in [Table 6](#) below and in [Figure 78](#) as follows.

### Sub net mask

Use the following sub net mask for **all** IP addresses: **255.255.252.0**.

### IP addresses (X.Y.Z.Q)

- Use the recommended IP addresses in the **X.Y.** positions of X.Y.Z.Q for both redundant networks; **172.16.Z.Q** for the primary network, and **172.17.Z.Q** for the secondary network.
- Due to the sub net mask value, select the **Z.** position of X.Y.Z.Q as a multiple of four. Choose two values between; 172.16.**0.Q**, 172.16.**4.Q**, 172.16.**8.Q**, 172.16.**12.Q** etc. up to 172.16.**124.Q**. Note that the **Z.** value must also be different for Control Network and Client/Server Network.
- Select the **Q** position of X.Y.Z.**Q** as a free serial number in the range 1 - 254, for each **node** on each separate network. Thus one serial **Q** number for the primary and the secondary network ports of each controller and PC.



*Table 6. Selected settings of the IP addresses with the 255.255.252.0 subnet mask.*

<b>Network</b>	<b>Controller_1 (X.Y.Z.Q)</b>	<b>Controller_2 (X.Y.Z.Q)</b>	<b>PC_1 with Control Builder M (X.Y.Z.Q)</b>	<b>PC_2 with OPC Server (X.Y.Z.Q)</b>
Primary Control Network	172.16.0.1	172.16.0.2		172.16.0.3
Secondary Control Network	172.17.0.1	172.17.0.2		172.17.0.3
Primary Client/Server Network			172.16.4.1	172.16.4.2
Secondary Client/Server Network			172.17.4.1	172.17.4.2

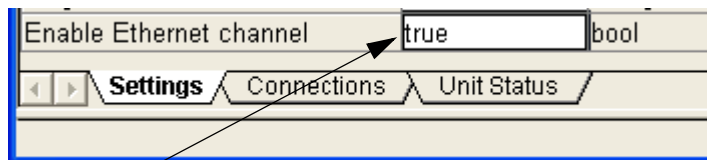
## Setup Using the IPConfig Tool

To get initial access to the PLCs, you must give them a first primary IP address.

1. Connect a PC, running the program *IPConfig* tool, to the PLC\_1 Tool port (via a serial cable).
2. Follow IPConfig online help instructions and set the primary IP address to 172.16.0.1 (see [Table 6](#)).
3. Repeat these steps for PLC\_2.

## Configure Controller Ports from Project Explorer

1. In the Project Explorer of PC\_1, configure your project with PLC\_1 and PLC\_2.
2. In PLC\_1, double-click Ethernet port number 1 to open its editor.
3. In the editor Settings tab, set the IP address parameter to 172.16.0.1 (see [Table 6](#)), and the IP sub net mask to 255.255.252.0. No other parameter setting is required.
4. Repeat step 2 and 3 for Ethernet port number 2 of PLC\_1, set its IP address to 172.17.0.1 (see [Table 6](#)).
5. From Ethernet port number 2, select **Settings** tab and set the **Enable Ethernet channel** parameter to true ([Figure 81](#)).



Change parameter from false to true!

*Figure 81. Hardware editor for Ethernet port No. 2 in Project Explorer. The parameter must be true for Network redundancy.*

6. Repeat step 2 to 5 for PLC\_2.

## Configure PC Ports in Windows XP Professional

For PC\_1 and PC\_2 you configure their Ethernet board ports by the following steps:

1. In PC\_1, select **Start>Control Panel**.
2. Select **Network and Internet connections**.
3. Select **Network Connections**.
4. In Network Connections, right-click Local Area Connection and select **Properties**. The Local Area Connection Properties dialog is displayed.
5. In the Connect Using list, select the Ethernet board that is to be connected to the primary network.
6. Select **Internet Protocol (TCP/IP)** and click **Properties**. The Internet Protocol (TCP/IP) Properties dialog is displayed.
7. Select **Use the following IP address** and enter the IP address (172.16.4.1 for the primary network in PC\_1, see [Table 6](#)), and the sub net mask 255.255.252.0. Click **OK**.
8. Repeat steps 1 to 7 for the secondary network (using the IP address 172.16.4.2).
9. Click **OK**.
10. For PC\_2, repeat steps 1 to 9 for **all four** Ethernet boards to be configured.
11. Click **OK**. The PC IP addresses and sub net masks have now been set for all Ethernet ports connected to the two networks.

## Download Project and Go Online

When all IP addresses and sub net masks are set, download your project and go online in the Control Builder. After a while, redundant network communication is enabled.

## Design

### IP Address

A communication channel IP address is a 32-bit word (4×8 bits) that can be entered as a string X.Y.Z.Q of four decimal numbers 0-255, separated by periods. The IP standard uses the terms *NetID* and *HostID*. The *sub net mask* specifies the boundary between the NetID part and the HostID part of the IP address (the zero bits indicate the HostID part). Depending on the value of X, IP addresses are divided mainly into three classes, A–C:

Table 7. IP address classes.

Class	Value of X	NetID	HostID	Possible host IP addresses	Default subnet mask
A	1-126	X	Y.Z.Q	X.0.0.1-X.255.255.254	255.0.0.0
B	128-191	X.Y	Z.Q	X.Y.0.1-X.Y.255.254	255.255.0.0
C	192-223	X.Y.Z	Q	X.Y.Z.1-X.Y.Z.254	255.255.255.0

The *Redundant Network Routing Protocol (RNRP)* developed by ABB handles alternative paths between nodes and automatically adapts to topology changes. RNRP uses more terms than the standard IP, namely *network area* and *node number*. By selecting 225.255.252.0 as the sub net mask, the last 10 bits constitute the node number (i.e. host ID, 0-1023). Note that the largest permitted node number is 500. The remaining NetID part is used for network ID (16 bits), local flag (1 bit), and network area number (5 bits). The last two bits of the network ID make up the path number, where 0 indicates the primary network and 1 the redundant secondary network.

Consequently, RNRP requires a different interpretation of the IP address to the IP standard.

It is recommended that the RNRP interpretation of the IP address be used. If the decimal numbers are converted to binary numbers, the address can be interpreted as follows:

XXXXXXXX.XXXXXXPP.LAAAAANN.NNNNNNNN

Each position represents a binary digit (0 or 1). The different parts of the address signify the following:

*Table 8. IP address converted into binary.*

Binary number	Number of bits	IP term	RNRP term
XXXXXXXX.XXXXXXPP	16	Network ID	Network ID
LAAAAA		Subnetwork ID	
PP	2		Path number
L	1		Local flag
AAAAA	5		Network area number
NNNNNNNNN	10	Host ID	Node number

The sub net mask sets the boundary between the host ID part and the sub net ID part and is selected as 11111111.11111111.11111100.00000000 (=255.255.252.0 in decimal notation).



The network ID must be identical for all nodes on the same network. It is recommended that an address be selected from the private IP address space, which has the following advantages:

- There is no requirement to apply to the licensing authorities for an IP address.
- Some protection is gained against illegal access, since private addresses are not permitted on the public Internet.
- The firm connection between IP and RNRP parameters reduces the risk of inconsistency.



The following primary network IP addresses are recommended (class B):

172.16.0.0  
 172.20.0.0  
 172.24.0.0  
 or 172.28.0.0

By converting the second decimal number into binary, you will find that the path number is 0 (see also [Table 6](#) and [Table 8](#)).

### Example 1:

Convert the sub net ID 11111111.11111111.11111100.00000000 to decimal.

By writing the first part in groups of four binary digits (1111), you will find that the hexadecimal equivalent is FF. The decimal equivalent is 255. The second part is identical, that is, 255. The third part 1111 1100 equals FC (hexadecimal) and 252 (decimal). The fourth part equals 0 in both decimal and hexadecimal notation. Consequently the sub net ID is written 255.255.252.0 in decimal notation.

### Example 2:

How is the IP address 192.168.255.25 written in binary notation?

It can be written C0.A8.FF.19 in hexadecimal form, which means 11000000.10101000.11111111.00011001 in binary notation.

## Separating Client/Server and Control Network

The Control Network must be protected from foreign traffic that can be a security risk and also cause undesired load on both the nodes and network. To avoid these risks the Control Network should be physically separated from the Internet and protected by servers and/or fire walls. In large configurations such separation may also be desirable between the Control Network and client/server networks.

### Client Node and Server Node

In a large network topology, only nodes with real-time communication requirements may be connected to the Control Network. *Client nodes* (such as engineering stations, operator stations, etc.) not belonging to the Control Network may go through a *server node* that performs authority control and can disable routing (transfers) to nodes on the Control Network.

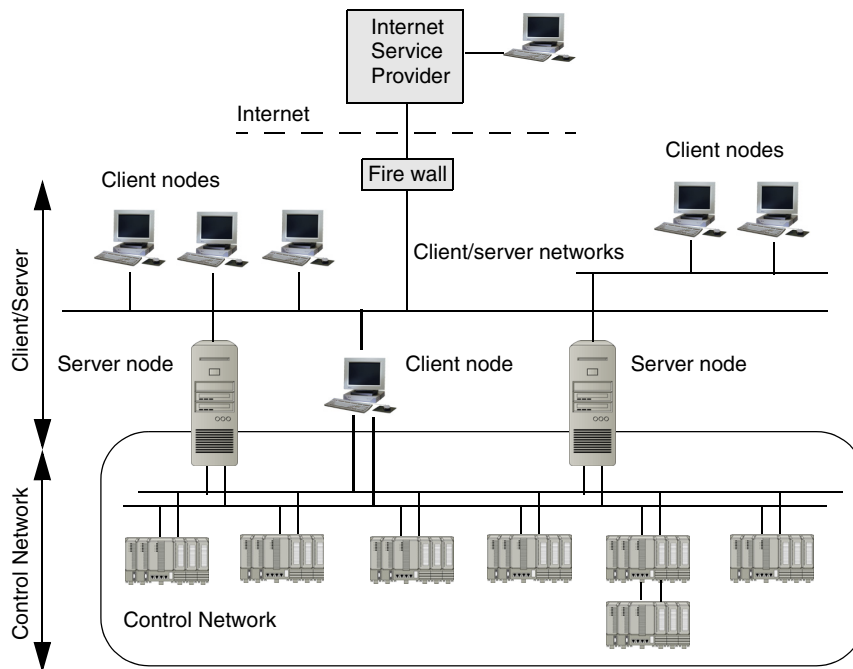


Figure 82. Separation of Control Network and Client/server networks.

It is, however, possible to have client nodes such as operator stations and engineering stations connected directly to the Control Network. If such a client needs access to the client/server network, a separate network connection is possible.



Traffic other than that between PLC Control Builder products may jeopardize performance.

Only IP traffic generated by Industrial<sup>IT</sup> products is allowed to reach the controllers on the Control Network.

## Summary of Configuration Steps

1. Specify network ID (IP address and sub net mask)
2. Split the control network into network areas
3. Specify local network areas
4. Specify network area numbers
5. Assign path numbers
6. Decide which devices should run the routing functionality
7. Assign IP addresses to each device
8. Assign redundant functionality to each system



---

## Appendix C Upgrade

PLC Control Builder is a new effective programming tool for creating control solutions, and you can upgrade your previous Control Builder M and Advant Control Builder solutions to a PLC Control Builder 4.0 project version. This appendix focuses mainly on upgrading previous projects to a 4.0 version.

However, upgrading to another PLC 4.x version needs much less work. Therefore, if you only need to upgrade projects within 4.x versions (4.0 to 4.1), you are advised to go straight to the subsection [Remove Products](#) on page 134, and then install the latest PLC Software and download new firmware to the PLC.

### Introduction

This release is **ONLY** compatible with Control Builder M (Basic, - Standard, - Professional) version 3.2/7 (or later 3.2 versions). Thus, previous versions like 3.x, 2.x or Advant Control Builder, must first be upgraded to 3.2/7 or later 3.2 versions.

PLC Control Builder AC 800M comes with features and performance similar to Control Builder M Professional 3.2 version.

An upgrade involves the following:

- Licenses, see page 130.
- Products, see page 131.
- Applications, see page 132.



Upgrading requires the plant to be shut down. To guarantee the functionality of the upgraded system, follow these upgrade instructions carefully and perform them in a well defined order.

## Licenses

Contact your sales representative to perform the license upgrade. Collect all Industrial IT licenses and have them available when contacting ABB. [Table 9](#) lists and describes the Control Builder license structure.

*Table 9. License Structure between version 3.2 and 4.0*

License in Version 3.2	License in Version 4.0
Advanced Process Control license	Obsolete - now included in CPU.
Basic Control Software license	
Binary Control Software license	
Control Builder M Basic	PLC Control Builder AC 800M
Control Builder M Professional	
Control Builder M Standard	
OPC Server for AC 800M/C license	OPC Server for AC 800M

## Products



All controllers, Control Builder and OPC Server nodes must be shut down during the upgrade. A plant shutdown is needed, because a software upgrade of the controller firmware and control software stops controller execution.

[Table 10](#) lists and describes specific considerations to take into account before and after upgrading.

*Table 10. Product Upgrade Considerations*

Issue	Considerations
Compatibility	Control Builder, AC 800M Firmware, and OPC Server for AC 800M must all be upgraded to the new version in order to work together. However, controller peer-to-peer communication is possible with other controllers running version 2.x/y or 3.x./y by means of Access Variables.
Privilege Handling	All privilege/user handling done in Control Builder M in the previous version (PrivUsers/PrivClasses) will be lost at the upgrade. The PLC Control Builder does not have any privilege handling.
AC 800C and Advant Controller 250 is no longer supported.	Projects including these controller types cannot be upgraded.

# Applications

## Saving Application and Configuration Data

The data described under the following subsections should be saved on a safe media before upgrading.

### Settings

It is very important to save Cold Retain values from the previous version on files, before beginning the upgrade.

1. From Control Builder M Menu bar, select: **Tools > Save ColdRetain Values**



Control Builder M must be in the Online mode in order to use the **Tools > Save ColdRetain Values** menu command.

2. Repeat the procedure for each Control Builder M project.  
Failure to make new files can cause some or all Cold Retain marked variables to revert to older values after upgrading.
3. Record the memory setting for OPC Server and Control Builder M found in the Setup Wizard for each product.
4. From the OPC Server Panel, select **File > Save Configuration**. The OPC configurations will be saved.

### Projects

Projects created with Control IT for AC 800M/C Version 3.2/7 (or later 3.2 version) can be used after the upgrade; however, in some cases minor modifications must be made.

- **Save Control Builder M Projects as Official Versions:** Before upgrading a project, save the entire project as an Official version. Unofficial parts of a project will not be upgraded. If the projects contain user-made HWD-files, they must also be saved. Save the Control Builder M project directory on a safe media; include new Cold Retain values files.
- A single library can not be upgraded by itself. To upgrade a library, include it in a dummy project.

- Acquired libraries that are not delivered together with Control Builder M should normally not be upgraded. Contact the vendor of the acquired libraries to get an updated version.

### OPC Server for AC 800M

Copy the following OPC Server files from the OPC Servers to safe media:

- systemsetup.sys, located in:  
`...\ABB Industrial IT Data\Control IT Data\  
OPC Server for AC 800M n.n`
- Configuration files (\*.cfg), located in:  
`...\ABB Industrial IT Data\Control IT Data\  
OPC Server for AC 800M n.n\Files`

### Control Builder M

The procedure for backing up Control Builder M projects is described in *Control Builder Online Help*. Choose to make a complete backup.

Standard libraries, and all other libraries, stored in the Control Builder M standard library path, typically:

```
...\Program Files\ABB Industrial IT\Engineer IT\  
Control Builder M [variant] 3.2\Libraries
```

or below, are not included in a project backup. The standard libraries will exist in new versions in the new system. Other libraries in that path or below must be handled manually. The best method is to save them to another path before making the backup.



Do not compress the backup.

## Remove Products

Stop third party OPC Clients and shut down Control Builder on all nodes.

In order to upgrade, the previous version or revision should be **removed** before installing the new version. If several products (that is, Control Builder M, and OPC Server) are installed on the same PC, all should be removed before the new versions are installed.

The following services must be manually stopped before the products are removed/updated:

- MMS Server for AC 800M,
- OPC Server for AC 800M,
- Tool Routing Service for AC 800M
- RNRP Service.



It also applies for any other service that automatically starts any of above services.

The RNRP Service should be stopped in the Windows Services Overview, which is found at **Start > Control Panel > Administrative Tools > Services**.



No existing projects will be deleted from disc when a previous Control Builder version is removed. The new Control Builder will be able to upgrade the projects left on disc.

## Install Products

Install the products by follow the instructions given in [Section 2, Installing PLC Software](#), (this manual).

## Restore Application and Configuration Data

### OPC Server for AC 800M

The memory setting for OPC Server needs to be set from the Setup Wizard. This is the value recorded under [Saving Application and Configuration Data](#) on page 132.

1. Manually restore the previously saved systemsetup.sys file to the following location:

```
ABB Industrial IT Data\Control IT Data\OPC Server for  
AC 800M 4.0
```

Restore the configuration files (\*.cfg) to the Files folder in the same location.

### PLC Control Builder

Do not start PLC Control Builder until Step 3.

The memory setting for PLC Control Builder needs to be set in the Setup Wizard. This is the value recorded under [Saving Application and Configuration Data](#) on page 132.

PLC Control Builder version 4.0 cannot be used for restoring a backup diskette that is made in a Control Builder M 3.2/x version. The files on a backup diskette can, however, be restored manually using Windows Explorer.

PLC Control Builder version 4.0 stores all source code as XML-code. An existing project and source code units (applications, libraries, and controller files) must be converted to the new source code format.

Saved Libraries stored on the Control Builder M standard library path, typically:

```
...\Program Files\ABB Industrial IT\Engineer IT\  
Control Builder M [variant] 3.2\Libraries
```

will not be found during the conversion procedure. Perform the following steps before the conversion if such libraries exist.

Move the libraries to the folder:

```
...\Program Files\ABB Industrial IT\Engineer IT\  
PLC Control Builder AC 800M\Libraries
```

A library will be converted only once. It is possible to upgrade several projects that include the same libraries, but the libraries will only be converted during upgrade of the first project.

1. Copy the Control Builder M projects.
  - a. If the projects are left on the hard drive after removing a previous Control Builder version, then upgrade from that location without moving the projects. If the projects are restored from a backup diskette then copy the projects to a directory located directly in the root directory on the hard drive. Example: **C:\MyTempUpgrade**  
Include all Cold Retain value files and any user-made HWD-files.



It is not possible to do an upgrade if the projects are copied to the PLC Control Builder project directory. The default PLC Control Builder project directory is:

```
...\ABB Industrial IT Data\Engineer IT Data\  
PLC Control Builder AC 800M\Projects
```

- b. Make sure that all files in the project directory are not marked as Read-only which can happen if the files have been stored on a CD-ROM.
  - Use Windows Explorer to select the folder.
  - Right-click **Properties** to open the Properties dialog.
  - Deselect the **Read-only** check box in the Attributes frame.
  - When the Confirm Attributes Changes dialog appears, select **Apply changes to this folder, subfolders and files**.
  - Click **OK** in the Confirm Attributes dialog and then again in the Properties dialog.
- c. If the projects contain user-made HWD-files, they must be converted with the GSD Import Tool Version 3.3. Refer to the Hardware Definition Files Syntax issue in [Table 11](#). Replace the old HWD-files with the new converted HWD-files and make sure that no files with the old format are left.



- d. If the projects contain S900 I/O, they must be converted with the S900 I/O Conversion tool. Refer to the S900 I/O Parameter Name Changes issue in [Table 11](#).
2. Upgrade the Control Builder M project.

Libraries, Applications, and Controllers originally placed in folders different from the project folder will not be upgraded if the project backup on safe media are restored using Windows Explorer.

The following manual adjustments to the project file (\*.prj) is needed in order to upgrade such projects correctly. Delete all directory references, other than those to *Libraries:\*.

**Before adjustments:**

```
FileUnits
( Application
  ( Name 'Application_1'
    Directory '' )
  Library
  ( Name 'AlarmEventLib'
    Directory 'Libraries:\' )
  Library
  ( Name 'IconLib'
    Directory 'Libraries:\' )
  Library
  ( Name 'MyOwnLibrary'
    Directory '..\..\MyOwnLibs\' )
  ControlSystem
  ( Name 'Controller_1'
    Directory '' ) )
```

**After adjustments:**

```
FileUnits
( Application
  ( Name 'Application_1'
    Directory '' )
  Library
  ( Name 'AlarmEventLib'
    Directory 'Libraries:\' )
  Library
  ( Name 'IconLib'
    Directory 'Libraries:\' )
  Library
  ( Name 'MyOwnLibrary'
    Directory '_' )
  ControlSystem
  ( Name 'Controller_1'
    Directory '' ) )
```

3. From the PLC Control Builder select **Tools > Maintenance > Upgrade Project**. A dialog window opens.
4. Click **OK**.
5. Browse to the project file (\*.prj) and click **Open**. This will upgrade the selected project.
6. Wait for the upgrade to finish.
  - a. During the upgrade process, some messages may be displayed. Read the messages and choose among the available options.
  - b. The upgrade may take up to 30 minutes. The PLC Control Builder may stop responding during that time. If the process is interrupted, it must be restarted. The Windows Task Manager can be used to supervise the completeness.



During the upgrade, if the mouse pointer, when moved over the PLC Control Builder Project Explorer, indicates it is busy, then the upgrade is still in progress.

7. Repeat [Step 3](#) to [Step 6](#) for all other projects.

## Downloading the Project

A project created in Version 3.2 can be used; however, minor modifications may be required. Minor changes need to be done if there is an issue in [Table 11](#) that are applicable for the particular application program.

1. Open the project.
2. Ignore the following warning messages that may be displayed in the message pane of the Project Explorer when opening the project.
  - At line xxx: Unknown parameter name: xxx
  - At line xxx: Parameter type mismatch: xxx

Make a dummy change in the hardware configuration for all hardware units if such messages are shown.

3. Modify the application program according to applicable issues in [Compatibility Issues](#) on page 141.

4. Upgrade controller firmware to the new version in order to be compatible with other products of version 4.0. Firmware in all communication interfaces must also be upgraded to the latest version.
5. Go online with *download* and answer any mismatch dialog with the corresponding new name, in order to retain any Cold Retain values (for example settings of PID loops etc.).



It is very important to complete the mismatch handling during the first download. If not, some or all Cold Retain marked variables will lose their saved values. See [Handling Mismatch for cold retain values](#) on page 140

### Handling Mismatch for cold retain values

Upgrade to version 4.0 may give mismatch for cold retain values.

- For example, the 3.2/x library SystemLib is nowadays divided into a System folder and a Basic library.
- Furthermore, the CommunicationLib from 3.2/x has been divided into several communication libraries e.g. MMSCCommLib, ModBusCommLib etc.

This means that PLC Control Builder will display the version mismatch dialog when downloading the project after upgrading to version 4.0. To prevent that cold retain values are lost during upgrading, study the solutions provided below.

Solutions:

- An Application not found mismatch dialog may warn for the Source code unit SystemLib when an upgraded project is downloaded for the first time. Click **Rename** in the dialog and choose to rename the SystemLib to BasicLib. If this does not work, then the application does not have any type that has been moved to BasicLib. Cancel the Rename dialog and select **Next Mismatch**. Refer to Control Builder Online Help for further information.
- If the version mismatch dialog displays 'CommunicationLib not found' rename the old CommunicationLib to any of the new divided communication libraries in the version mismatch dialog.

6. From the hardware tree in Project Explorer, select the controller and right-click **Properties > Heap Utilization**. A Heap Utilization dialog opens.
7. Check the spare memory needed for online changes in each controller.
8. Repeat these steps for all other projects.

### Compatibility Issues

Table 11 lists compatibility issues between PLC Control Builder and Control Builder M and corresponding descriptions and solutions.

Table 11. Compatibility Issues

Issue	Description/Solution
System Library	<p>The SystemLib is split into two different libraries:</p> <ul style="list-style-type: none"> <li>• System: Includes firmware functions and data types that are defined within firmware. This library is not version handled.</li> <li>• BasicLib: Includes the rest of the former SystemLib. This library uses version handling.</li> </ul> <p>BasicLib must always be inserted into the project.</p> <p>If the version mismatch dialog displays 'SystemLib not found' rename SystemLib to BasicLib in the mismatch dialog. The dialog may also display mismatches for several basic data types e.g. <i>RealIO</i>, <i>BoolIO</i> etc., which are nowadays located in the System folder. Just select 'Next mismatch'.</p> <p>Refer to <i>Control Builder Online Help</i> for further information.</p>
Process Object Libraries	<p>This version contains enhanced versions (2.x/y) of the ProcessObjBasicLib and ProcessObjExtLib libraries. These are not fully compatible with the previous versions. The previous versions (1.x/y) are included for upgrade reasons. Do not use them for new applications.</p>

Table 11. Compatibility Issues (Continued)

Issue	Description/Solution
<p>Communication Library</p>	<p>The <i>CommunicationLib</i> is split into the following smaller libraries:</p> <ul style="list-style-type: none"> <li>• MMSCommLib</li> <li>• FFH1CommLib</li> <li>• SattBusCommLib</li> <li>• ModBusCommLib</li> <li>• COMLICommLib</li> <li>• S3964CommLib</li> <li>• ModemCommLib</li> </ul> <p>They are fully compatible with the former <i>CommunicationLib</i>.</p>
	<p><i>CommunicationLib</i> (Version 1.0/0) is still included for upgrade reasons, but will be removed in future versions.</p>
	<p>The following steps are necessary when replacing the old library:</p> <ol style="list-style-type: none"> <li>1. Open the project where <i>CommunicationLib</i> is still connected to the applications/libraries.</li> <li>2. Insert needed new communication libraries to the project, for example <i>MMSCommLib</i>, if the MMS protocol is used.</li> <li>3. Connect the inserted libraries to the applications/libraries in the project.</li> <li>4. Disconnect <i>CommunicationLib</i> from the applications/libraries. All Function Blocks will automatically be replaced.</li> </ol>
	<p>Perform the following additional steps for each instance of <i>CCToFF</i>, <i>FFToCC</i>, <i>CCToMMS</i>, or <i>MMStoCC</i> in the applications/libraries:</p> <ol style="list-style-type: none"> <li>1. Connect the <i>FFH1CommLib</i> (<i>CCToFF</i>, <i>FFToCC</i>) or <i>MMSCommLib</i> (<i>CCToMMS</i>, <i>MMStoCC</i>).</li> <li>2. Mark the instance in Project Explorer.</li> <li>3. Right-click and select to replace the Type.</li> <li>4. Disconnect the <i>CommunicationLib</i>.</li> <li>5. Select the Type from the new library.</li> <li>6. Save and close.</li> </ol>

Table 11. Compatibility Issues (Continued)

Issue	Description/Solution
Serial Library	<p>SerialLib has been renamed to SerialCommLib.</p> <p>Perform the following steps:</p> <ol style="list-style-type: none"> <li>1. Open the project where SerialLib is still connected to the applications/libraries.</li> <li>2. Insert the SerialCommLib library to the project.</li> <li>3. Connect the SerialCommLib to the applications/libraries in the project.</li> <li>4. Disconnect SerialLib from the applications/libraries. All Function Blocks will automatically be replaced.</li> </ol>
Library Dependencies	Circular dependencies between libraries is no longer permitted. The use of circular dependencies causes compilation errors.
ReallO Status	The status component of the ReallO data type has changed data type from dint to dword. In most cases this change will be taken care of automatically, but if the status component has been referred to directly in application code, a manual change is necessary.
Duplicate names	<p>PLC Control Builder has a stricter naming rule than in the previous versions. The instance names for variables, function blocks, control modules etc., have to be unique within the object were the instance is declared e.g. program or control module. When upgrading a project from an older version and then trying to download there may be error messages during compilation like 'The variable name &lt;name&gt; is not unique'.</p> <p>Change the duplicate instance name so that they are unique.</p>
Hardware Definition Files Syntax	Version 4.0 requires that used HWD-files are of a newer syntax version. The new version (3.3) of the GSD Import Tool has an option to transform older HWD-files to the syntax that is required by Version 4.0. This is only applicable for user-made HWD-files, all standard HWD-files are updated. For detailed information, refer to the Update HWD-files online help topic in the GSD Import Tool.

Table 11. Compatibility Issues (Continued)

Issue	Description/Solution
<p>User-Defined Hardware Definition Files when 61131-application variables of type dInt and dWord are written from the application to I/O-channels.</p>	<p>The behavior has been undefined when the 61131-variable has a very large value (larger than the maximum value possible to write on a specific hardware unit). The very large value has really been truncated, a hardware unit receiving 8-bit data has received the 8 lowest bits in the dInt or dWord.</p> <p>The behavior is redefined as follows:</p> <ul style="list-style-type: none"> <li>• The hardware unit will be written with the largest, which is different depending on hardware, possible value if 61131-variable has a very large positive value.</li> <li>• The hardware unit will be written with the largest, which is different depending on hardware, negative value if 61131-variable has a very large negative value.</li> </ul>
<p>S900 I/O Parameter Name Changes</p>	<p>S900 I/O parameter names have been changed, which will result in loss of configuration data when upgrading a project containing S900 I/O modules.</p>
	<p>Error messages will be given during upgrade of the project if it has not been converted.</p>



Table 11. Compatibility Issues (Continued)

Issue	Description/Solution
S900 I/O Parameter Name Changes (cont)	<p>Before migrating a project, repeat the following steps for all *.con files in the project that may contain S900 I/O modules.</p> <ol style="list-style-type: none"> <li>1. Locate:            CBMconS900Conv.exe  in the <b>tools\ABB\CBMconS900Conv</b> directory on the CD.</li> <li>2. Open a command prompt window and change to the directory where the CBMconS900Conv.exe is present:            ...&gt; CD Z:\tools\abb\CBMconS900Conv\</li> <li>3. Enter CBMconS900Conv [path to project folder]ControllerName.con. For example:            CBMconS900Conv ...\....\Projects\MyProj\MyController.con.</li> <li>4. Press <b>Enter</b>. The MyController.con file will be renamed to MyController.bak, and a new converted MyController.con file will be generated. A message that the file was converted successfully will appear.</li> <li>5. Repeat Step 3 and Step 4 for all *.con files that includes S900 I/O modules.</li> </ol>
Metafiles Locations	<p>These files are not relocated on safe media during the source code conversion, which means that the image selectors will fail if the paths to the image files are specified relative to the project directory. If this happens; copy the files so that the used relative path points to the files, or change the used path to explicitly point to the image files (for example ...\ImageDirectory\ImageName.wmf).</p>
	<p>Windows metafiles (wmf) or enhanced metafiles (emf) that are used by image selectors in Control Module Diagrams are locally stored on disk and need to be present on all PLC Control Builder nodes. Only those with wmf and emf images included in Control Module diagrams will be affected.</p>
CI840	CI840 must be upgraded to firmware Version 3.0/0 or later.

Table 11. Compatibility Issues (Continued)

<b>Issue</b>	<b>Description/Solution</b>
DP820 Scaling	DP820 frequency scaling has changed. The default setting before was 0.0 to 1500000.0 Hz. It should now be -15.00000.0 to +1500000.0 Hz. DP820 will show the wrong frequency value if this change is not done.
Internal MMS Communication between applications residing in the same controller.	The behavior of internal communication between two applications in the same controller has changed. This kind of communication is now asynchronous; all data may no longer always be received in the same scan.

---

## Appendix D Communication Cables

Serial communication between Control Builder and the PLC 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 PLC controller COM4 port. The [Figure 83](#) illustrates the TK212A pin-out configuration.

### Connecting Control Builder PC to a PLC

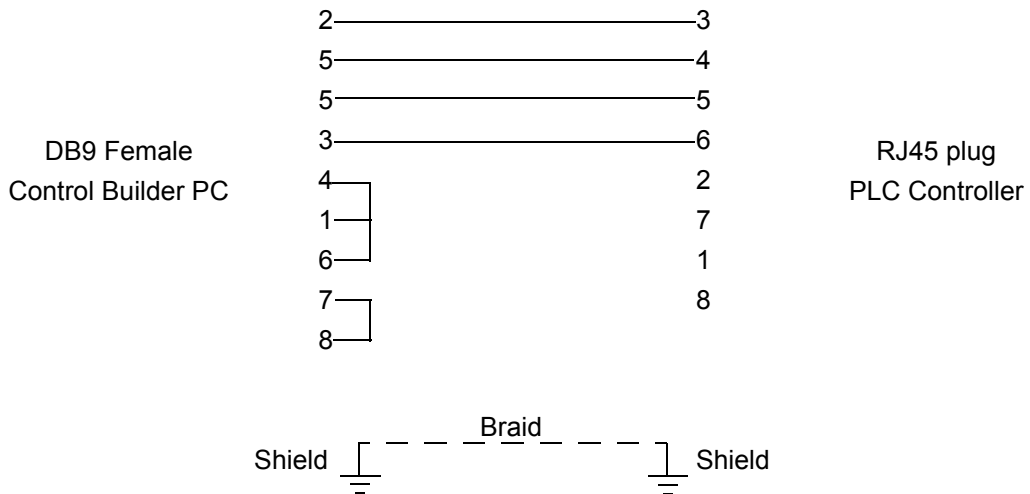


Figure 83. Cable TK212A Pinout configuration.



---

# Appendix E Programming Languages

PLC Control Builder provides five different programming languages according to IEC 61131-3. These are Function Block Diagram (FBD), Structured Text (ST), Instruction List (IL), Ladder Diagram (LD) and Sequential Function Chart (SFC). The specific rules and syntax of the programming languages will not be discussed in detail in this manual. Please refer to the Control Builder Online Help.



SFC Viewer is not supported by PLC Control Builder AC 800M.

## General

The IEC 61131-3 standard defines five of the most commonly used programming languages on the market. Depending on previous experience, programmers often have their own personal preference for a certain language. All the languages have advantages and disadvantages, and no single one of them is suitable for all control tasks. We start with three basic statements and then proceed to some important characteristics of each language.

- In small applications with relatively few logical conditions, the demand for good structure and re-use of code is not as great as in larger systems.
- ST and IL are textual languages, while FBD, LD, and SFC are based on graphical metaphors.
- LD and IL are not as powerful as ST or FBD.



Note that the definition of function block is allowed in all five languages, not only in FBD. A function block is a method of encapsulating the code in a “black box” with inputs and outputs.

All instructions and functions viewed in this section are explained in detail in Control Builder online help.

For more information on the different languages, see the manual *IEC 61131 Control Languages, Introduction*.

Some important characteristics of the languages are listed in the table below.

*Table 12. PLC Control Builder programming languages.*

Language	Function
Function Block Diagram (FBD)	A graphical language for depicting signal and data flows through function blocks and re-usable software elements. Function blocks and variables are interconnected graphically, which makes the resulting control diagrams easy to read.
Structured Text (ST)	A high-level programming language. ST is highly structured and has a comprehensive range of constructs for assignments, function/function block calls, expressions, conditional statements, iterations, etc.  It is easy to write advanced, compact, but clear ST code, due to its logical and structured layout.
Instruction List (IL)	A traditional PLC language. It has a structure similar to simple machine assembler code.
Ladder Diagram (LD)	Ladder diagram (LD) is a graphical language based on relay ladder logic.
Sequential Function Chart (SFC)	Sequential function chart (SFC) is a graphical language for depicting the sequential behavior of a control program.

---

## Appendix F Glossary

The following is a list of terms associated with PLC Control Builder that you ought to be familiar with. The list contains some terms and abbreviations that are unique to ABB or have a usage or definition that is different from standard industry usage.

Term/Acronym	Description
Application	The application contains the program code to be compiled and downloaded for execution in the controller. Applications are displayed in the Project Explorer.
Control Builder	A programming tool with a compiler for control software. Control Builder is accessed through the Project Explorer interface.
Control Module (Type)	A program unit that supports object-oriented data flow programming. Control modules offer free-layout graphical programming, code sorting and static parameter connections. Control module instances are created from control module types.
Firmware	The system software in the PLC.
Hardware Description	The tree structure in the Project Explorer, that defines the hardware's physical layout.
IEC 61131-3	A standard for control system languages defined by the IEC.
Industrial <sup>IT</sup>	ABB's vision for enterprise automation.
Industrial <sup>IT</sup> 800xA System	A computer system that implements the Industrial IT vision.

Term/Acronym	Description
Interaction Window	A graphical interface used by the programmer to interact with an object. Available for many library types.
MMS	<p>Manufacturing Message Specification, a standard for messages used in industrial communication.</p> <p>This is the application layer used within MAP (Manufacturing Automation Protocol), a specification for open communication based on the OSI model.</p>
OPC	OLE for Processing Control.
OPC/DA	An application programming interface defined by the standardization group OPC Foundation. The standard defines how to access large amounts of real-time data between applications. The OPC standard interface is used between automation/control applications, field systems/devices and business/office application.
Process Object	A process concept/equipment such as valve, motor, conveyor or tank.
Project Explorer	<p>The Control Builder interface. Used to create, navigate and configure libraries, applications and hardware.</p> <p>All objects such as data types, functions and function block types can be selected and displayed in an editor. The software and hardware is configured in the Project Explorer.</p>
RNRP	Redundant Network Routing Protocol.



<b>Term/Acronym</b>	<b>Description</b>
Security	<p>Security controls a user's authority to perform different operations on (Aspect) Objects, depending on several parameters:</p> <ul style="list-style-type: none"><li data-bbox="644 389 1268 415">• The user's credentials, as provided by Windows</li><li data-bbox="644 428 1310 582">• The node where the user is logged in. This makes it possible to give a user different authority depending on where he/she is located, for example close to the process equipment, in a control room, or at home accessing the system through Internet.</li><li data-bbox="644 596 1225 654">• The operation the user wants to perform the operation on.</li></ul>
Type	<p>A type solution that is defined in a library or locally, in an application. A type is used to create instances, which inherit the properties of the type.</p>



---

# INDEX

## A

AC 800M  
  cable configuration 147  
  License upgrade  
    SB 2.1/2 to SV 4.0 130  
  Product upgrade  
    SB 2.1/2 to SV 4.0 131  
AC800M 36  
Access Variables 41  
addresses  
  convert 125  
Alarm and Event Library 16  
analysis  
  test mode 64  
application 35  
Applications 39

## B

Basic Library 16, 49

## C

Cable Configuration  
  AC 800M 147  
Check 43  
Check icon 41  
class A  
  IP address 124  
class B  
  IP address 124  
class C  
  IP address 124  
client  
  node 127  
client/server networks 127  
code pane 50

Cold Retain 132  
Communication Library 16  
configuration  
  MMS 128  
Connected Libraries 40  
Contents Topic 24  
Context-Sensitive Help 24  
Control Builder M 129  
Control builder M  
  Copying files to safe media  
    SB 2.1/2 to SV 4.0 133  
Control Module Types 41  
control modules 18  
Control Network 127  
Controllers 39  
convert  
  addresses 125  
Copying files to safe media  
  Control builder M  
    SB 2.1/2 to SV 4.0 133  
  OPC server for AC 800M  
    SB 2.1/2 to SV 4.0 133  
counters 53  
CPU unit 41  
cross-over 88

## D

Data Types 41  
declaration pane 50  
Description 43  
Difference Report 64 to 65, 92  
Downloading  
  via Ethernet 89

- E**
- editor
  - programs 50
- EmptyProject 36
- Enable Ethernet channel 122
- Ethernet
  - download via 89
  - ports 90
  - set IP address 90
- Ethernet cable 88
  
- F**
- FBD 149
- File Location
  - select 103
- File server 105
- Firmware
  - upgrade via serial line 81
- firmware functions 40
- Firmware Version 83
- Forced 94
- Function Block Types 41
  
- G**
- GSD Import Tool 136
  
- H**
- hardware address 76
- Hardware Definition Files 136
- Heap 100
- Heap Utilization 141
- host ID
  - IP address 125
- HostID 124
- HWD-files 136
  
- I**
- I/O Address 80
- I/O channels 77
  
- Icon library 49
- IEC 61131 Control Languages 13
- IL 149
- implicit IP addressing 117
- Index 24
- index
  - online help 24
- Industrial IT licenses 130
- IP address 84, 86
  - class A 124
  - class B 124
  - class C 124
  - host ID 125
- IPConfig 84
  
- K**
- Keyword Search 24
  
- L**
- Language
  - select 102
- LD 149
- Libraries 39
  - AlarmEventLib 16
  - BasicLib 16, 49
  - CommunicationLib 16
  - Control Library 16
- local flag 125
- Local System Account 113
  
- M**
- manuals 21
- Memory 100
- Message 43
- message pane 50, 64
- mismatch 140
- MMS
  - configuration 128
- ModuleBus 74
- multi-user configuration 20

**N**

NetID 124  
 network area 124 to 125  
 network ID 125  
 node  
   client 127  
   server 127  
 node number 124  
   RNRP 125

**O**

online analysis 91 to 92  
 Online Documentation 21  
 Online Help 21  
 online help  
   index 24  
   text search 25  
 Online Manuals 21  
 online mode 91 to 92  
 OPC panel 33  
 OPC Server 27  
 OPC server for AC 800M  
   Copying files to safe media  
     SB 2.1/2 to SV 4.0 133

**P**

parameters  
   RNRP 124  
 path number  
   RNRP 125  
 Permission 105  
 permission Full Control 106  
 PLC Control Builder 27  
 PLC control builder version 81  
 PLC firmware 81  
 ports  
   Ethernet 90  
   IP address 90  
 program 35  
 program editor 50

project 35  
 project documentation 21  
 Project Explorer 15  
 Project folder 20, 103

**R**

real-time communication 127  
 refresh 37  
 RNRP  
   node number 125  
   parameters 124  
   path number 125  
 RNRP Service 134  
 RNRP Setup Wizard 118  
 RNRP-icon 118

**S**

SB 2.1/2 to SV 4.0  
   AC 800M considerations  
     License upgrade 130  
     Product upgrade 131  
   Copying files to safe media  
     Control builder M 133  
     OPC server for AC 800M 133  
 search  
   online help 25  
 Selecting  
   file location 103  
   language 102  
 server  
   node 127  
 Service Account 112 to 113  
 session log file 32  
 Settings tab 90  
 Setup Wizard 97  
 SFC 149  
 single-user configuration 27  
 SoftController 36  
 software model 73  
 ST 149

standard libraries 16  
straight-through 88  
structuring code 55  
subnetwork ID 125  
System folder 36

## T

Tasks 41  
templates 36  
test mode 64  
    analysis 64  
This Account 113  
Timers 53  
TK212A 82  
TK212A cable 147

## U

UNC path 107, 111  
Upgrade firmware via serial line 81

## V

variable conditions 67

## X

XML-code 135





3BSE039838R201. Printed in Sweden May 2005  
Copyright © 2005 by ABB. All Rights Reserved  
® Registered Trademark of ABB.  
™ Trademark of ABB.

<http://www.abb.com>

Automation Technology Products  
Wickliffe, Ohio, USA  
[www.abb.com/controlsystems](http://www.abb.com/controlsystems)

Automation Technology Products  
Västerås, Sweden  
[www.abb.com/controlsystems](http://www.abb.com/controlsystems)

Automation Technology Products  
Mannheim, Germany  
[www.abb.de/controlsystems](http://www.abb.de/controlsystems)