

ProLinux

7000-ADM

AppSrvCE

Application Development Module

User Manual

February 20, 2007



ProSoft
TECHNOLOGY

Please Read This Notice

Successful application of this module requires a reasonable working knowledge of the ProLinx Module, its connected devices, and the application in which the combination is to be used. For this reason, it is important that those responsible for implementation satisfy themselves that the combination will meet the needs of the application without exposing personnel or equipment to unsafe or inappropriate working conditions.

This manual is provided to assist the user. Every attempt has been made to assure that the information provided is accurate and a true reflection of the product's installation requirements. In order to assure a complete understanding of the operation of the product, the user should read all applicable documentation on the operation of the connected devices.

Under no conditions will ProSoft Technology be responsible or liable for indirect or consequential damages resulting from the use or application of the product.

Reproduction of the contents of this manual, in whole or in part, without written permission from ProSoft Technology is prohibited.

Information in this manual is subject to change without notice and does not represent a commitment on the part of ProSoft Technology Improvements and/or changes in this manual or the product may be made at any time. These changes will be made periodically to correct technical inaccuracies or typographical errors.

Your Feedback Please

We always want you to feel that you made the right decision to use our products. If you have suggestions, comments, compliments or complaints about the product, documentation or support, please write or call us.

ProSoft Technology

1675 Chester Avenue, Fourth Floor
Bakersfield, CA 93301

+1 (661) 716-5100

+1 (661) 716-5101 (Fax)

<http://www.prosoft-technology.com>

Copyright © ProSoft Technology, Inc. 2000 - 2007. All Rights Reserved.

7000-ADM User Manual

February 20, 2007

PSFT.AppSrvCE.ProLinx.UM.07.02.20

Important Installation Instructions

Power, input and output wiring must be in accordance with Class I, Division 2 wiring methods – Article 501-4 (b) of the National Electrical Code, NFPA 70 and in accordance with the authority having jurisdiction. The following warnings must be heeded:

- a** WARNING – EXPLOSION HAZARD – SUBSTITUTION OF COMPONENTS MAY IMPAIR SUITABILITY FOR CLASS I, DIV. 2;
- b** WARNING – EXPLOSION HAZARD – WHEN IN HAZARDOUS LOCATIONS, TURN OFF POWER BEFORE REPLACING OR WIRING MODULES, and
- c** WARNING – EXPLOSION HAZARD – DO NOT DISCONNECT EQUIPMENT UNLESS POWER HAS BEEN SWITCHED OFF OR THE AREA IS KNOWN TO BE NONHAZARDOUS.
- d** "THIS DEVICE SHALL BE POWERED BY CLASS 2 OUTPUTS ONLY."

Contents

PLEASE READ THIS NOTICE	2
Your Feedback Please	2
Important Installation Instructions	3
1 INTRODUCTION	7
1.1 Introducing the System	7
1.1.1 AppSrv CE Features	7
1.1.2 AppSrv_Full WinCE 4.2 Platform Release Notes	10
1.2 Important User Information	18
1.2.1 Preventing Damage from Electrostatic Discharge	18
1.2.2 Surge Protection for AppSrvCE Power Input.....	18
1.2.3 Wiring Requirements	19
1.2.4 Controller Spacing	20
1.2.5 Grounding the Controller.....	21
1.3 Environmental Specifications	22
2 INSTALLING APPSRVCE	23
2.1 Unpacking the Base Unit	23
2.2 Viewing the Base Unit	24
2.3 Inserting/Removing a CAM	24
2.4 Mounting the Unit	25
2.4.1 Panel Mount.....	26
2.4.2 DIN-rail.....	27
2.4.3 Installing the CAM.....	30
2.4.4 Removing the Base Unit from the DIN Rail	31
2.5 Power Supply Requirements	31
2.5.1 For Rockwell Automation Flex I/O	32
2.5.2 For Rockwell Automation Compact I/O.....	32
2.6 Wiring the Base Unit Power Supply	32
3 EZ DESIGN STUDIO	33
3.1 What is EZ Design Studio?	33
3.2 Flowchart	34
3.3 EZ Design Studio Software Components	34
3.3.1 EZ Design Studio Features	35
3.3.2 Platform Requirements	37
3.4 Installing EZ Design Studio	37
3.4.1 Post-Installation	37
3.4.2 Runtime Connection	38
3.4.3 Where to Get More Information	38
3.5 EZ Design Studio Applications	38
4 APPLICATION DEVELOPMENT OVERVIEW	39
4.1 Configuring and Programming	39
4.2 Running and Operating	39
4.3 Procedures	40
4.3.1 Embedded (Physical Control Node)	40

4.3.2	EZ Manager.....	40
4.3.3	EZ Studio.....	40
5	RUNNING EZ DESIGN STUDIO.....	43
5.1	Planning	44
5.2	Startup and System Configuration.....	44
5.3	Logic Control Programs.....	45
5.4	Scheduling Program Execution.....	45
5.5	Building and Downloading.....	45
5.6	Running and Monitoring	46
5.7	Using the Watch Window	46
6	EZ DESIGN STUDIO TUTORIAL	47
6.1	EZ Studio	47
6.2	Configuring the Node	49
6.3	Creating a Simple Program.....	49
6.4	Creating a Task	52
6.5	Building, Downloading, and Running the Program.....	54
6.6	Monitoring the Program	54
6.7	Conclusion.....	57
	SUPPORT, SERVICE & WARRANTY.....	59
	Module Service and Repair.....	59
	General Warranty Policy – Terms and Conditions	60
	Limitation of Liability.....	61
	RMA Procedures	61
	INDEX.....	63

1 Introduction

In This Chapter

- Introducing the System 7
- Important User Information 18
- Environmental Specifications 21

1.1 Introducing the System

The AppSrvCE is a modular, PC-based controller designed for stand-alone monitoring and control applications. The system has:

- A Base Unit which houses the CPU and supports direct connection for supported Rockwell Automation I/O. Refer to Specifications section in this document for a list of base unit catalog numbers and configurations.
- A Custom Application Module (CAM), not considered part of base unit. CAMs allow you to add serial ports, PC card functionality, and other options.
- Embedded I/O support
- Rockwell Automation power supply, catalog number 1794-PS1, and auxiliary Class 2 supplies as needed (ordered separately)
- Optional Rockwell Automation I/O (ordered separately)

1.1.1 *AppSrv CE Features*

The following table contains a list of features in Platform Builder 4.2 that may be selected for a CE build. This is not an exhaustive list; some features that are obviously required or not applicable to the AppSrv CE platform have been omitted.

Notes '•' means that this feature is included in the indicated platform.

WinCE 4.2 Feature	AppSrv_Full
Active Template Library (ATL)	•
.Net Compact Framework	•
SQL Server 2000 .NET Data Provider	
SQL Server CE 2.0 .NET Data Provider	
Smart Device Authentication Utility	
C++ Exception Handling	•
Run-time Type Information	•
Standard IO	•
Standard IO ASCII	•

WinCE 4.2 Feature	AppSrv_Full
DCOM	•
COM Storage	•
DCOM Remote Access	•
Device Management Client	
SNMP	
LDAP	
MSMQ	•
SRMP	
MFC	•
OBEX Server	
OBEX Client	
POOM API	
SOAP Toolkit Client	•
SOAP Toolkit Server	•
SQL Server CE 2.0	•
Standard SDK	
XML Core Services and DOM	•
XML HTTP	•
XML Query Languages (XQL)	•
XML SAX	•
XML Stylesheet Language Transformations (XSLT)	•
XML MIME Viewer	
ActiveSync	•
CAB File installation/uninstaller	•
Excel Viewer	•
Image Viewer	•
PDF Viewer	•
PowerPoint Viewer	•
Word Viewer	•
Solitaire	
Help	
Inbox	
Remote Desktop Client	•
Terminal Emulator	•
Windows Messenger	
WordPad	•
Wired LAN	•
Wireless LAN	
IrDA (PAN)	
Bluetooth	
RAS/PPP	•

WinCE 4.2 Feature	AppSrv_Full
PPPoE	•
TAPI w/Unimodem	•
PPTP	•
L2TP/IPSec	•
Extensible Authentication Protocol	
Internet Connection Sharing	•
Firewall	
Network Bridging	•
Network Utilities	•
RTC Client API	
TCP/IP	•
TCP/IP v6	
UPnP Control Point API	•
UPnP Device Host API	
Windows Networking API/Redirector	•
IP Helper API	•
Core Server Support	•
FTP Server	•
RAS/PPTP Server	
SNTP Service	•
Print Server	
Telnet Server	•
Web Server	•
ASP	•
Device Management ISAPI	•
Web Server Administration ISAPI	•
Toolhelp API	•
Fiber API	•
Remote Display Application (CERDISP)	•
Serial Port Support	•
USB Host Support	•
USB HID Class Driver	•
USB Printer Class Driver	•
USB Storage Class Driver	•
USB Remote NDIS Class Driver	
FAT File System	•
TFAT File System	•
FormatMessage API	
Internet Explorer 6.0	•
Pocket Internet Explorer	
Jscript 5.5	•

WinCE 4.2 Feature	AppSrv_Full
VBScript 5.5	•
Kerberos Authentication	
NTLM Authentication	•
SSL/TLS Authentication	•
CryptoAPI 1.0	•
CryptoAPI 2.0	•
Command processor	•
Console window	•
Accessibility	
Software input panel	
Platform Manager TCP/IP transport	•
Platform Manager KITL transport	•
Platform Manager ActiveSync transport	
Geode MediaGX display driver	•
Keyboard/mouse driver	•
DP83815 Ethernet driver	•
16550 Serial port driver	•
ATAPI IDE disk driver	•
DiskOnChip driver	•

1.1.2 *AppSrv_Full WinCE 4.2 Platform Release Notes*

Version 1.23

December 21, 2006

Introduction

This document provides information about this release of the Windows CE 4.2 operating system for AppSrv.

Important: This version includes a registry lock feature for improved reliability. The registry must be unlocked before any changes will be saved to the disk. Refer to the Registry Lock section for more information.

Supported Devices

The following support is present in this release:

- 256Mb RAM (maximum)
- Dual DP83815 Ethernet controllers
- Geode MediaGX Video controller
- USB Mouse and Keyboard
- PS/2 Mouse and Keyboard
- CompactFlash IDE disk
- M-Systems Disk On Chip
- COM1 and COM2 serial ports

- Auxiliary hardware timer
- 512Kbyte battery-backed Static RAM
- 1769 IO (Leo ASIC)
- Flex IO (Serbus ASIC)
- Geode watchdog
- High temperature sensor
- Front panel LED indicators
- DeviceNET CAM
- Quad Serial CAM
- Power fail handling
- Embedded IO

CE Components

The following CE components are included in this release (refer to the separate document **AppSrv CE Features** for more information):

- Internet Explorer 6.0
- HTTP Server
- FTP Server
- Telnet Server
- SNTP Service
- Excel, PowerPoint, Word, PDF, and Image file viewers
- ActiveSync
- CAB file installer
- WordPad
- Remote desktop client
- Terminal emulator
- .NET Compact Framework

Booting WinCE

The AppSrv BIOS supports booting to WinCE directly, or WinCE may be booted from MS-DOS using the LOADCEPC.EXE utility. The BIOS may be configured to boot from the CompactFlash (CF) or the Disk On Chip (DOC).

Follow these steps to configure the system to boot WinCE directly from the BIOS:

1. Copy the NK.BIN file to the root directory of the boot disk (must be FAT format).
2. Run BIOS Setup (press Del key immediately after reset to enter Setup) and set the BIOS Boot Method on the Basic CMOS Configuration page to 'Windows CE'.
3. Save the new settings and exit Setup.

Display Properties

The video display is configured for a resolution of 800x600, with a color depth of 64K (16bpp).

Storage Devices

Two types of storage devices are supported: CompactFlash cards, and a 32Mb onboard M-Systems DiskOnChip. The CF card will appear at IDE Disk1, and the DOC at \DiskOnChip.

Registry Persistence

The WinCE registry is restored from disk during startup, and resides in RAM while the system is running. In order for registry changes to be preserved after a system reboot, registry changes must be flushed to disk. A background thread called SaveReg has been implemented to periodically flush registry changes to disk.

By default, SaveReg runs every 30 seconds and has a priority level of 254. The interval and priority may be modified by editing the Period and Priority values of the following registry key:

```
HKEY_LOCAL_MACHINE\Software\Online Development\SaveReg
```

The Period is a DWORD value that should be set to the number of seconds between registry flushes. The Priority is a DWORD value that is used to set the priority level of the SaveReg thread.

The registry is saved on the DiskOnChip in the cesystem folder.

Note: SaveReg only runs if the registry is unlocked.

Registry Lock

When WinCE boots, device information is written to the registry and these changes are flushed to the registry hive on the boot disk. Therefore, every time AppSrv boots, WinCE re-writes the registry hive on disk. If system power is interrupted during this write, there is a small chance that the registry and/or file system may be corrupted. Beginning with release version v1.14, the AppSrv WinCE platform includes a feature to prevent this potential problem. The registry lock prevents writes to the registry hive on disk except when performing configuration.

In order to perform system configuration, the registry must be unlocked. The state of the registry lock can only be changed when the system is booted. Two utilities are provided to manage the registry lock: RegSave and TCU. Each of these utilities will warn the user if the registry is currently locked, and provide an option to unlock the registry at the next system boot.

By default, the registry is locked. RegSave or TCU can be used to temporarily unlock the registry; i.e., the registry will be unlocked for the next reboot, but will revert to the locked state at subsequent reboots.

If necessary, the registry lock feature can be disabled. This is not recommended, since it increases the chance of file system corruption as previously described. However, if losing system power suddenly is not a problem, for example, a UPS is being used, then this option may increase system ease-of-use, especially during development and configuration. To disable the registry lock, set the

DWORD value RegistryLock under the key
HKEY_LOCAL_MACHINE\Software\Online Development\SaveReg to 0.

The Utility API includes functions that can be used by an application to manipulate the registry lock. Refer to the Utility API section for more information.

Utility API

The Utility API contains several functions that applications can use to simplify tasks such as unlocking the registry, flushing registry changes to disk, and rebooting the system. To use the Utility API, include the header file **ceutilapi.h** and link with the import library **ceutilapi.lib**.

The following functions are included in the Utility API:

```
int UTL_IsRegistryUnlocked();
```

This function returns the status of the registry lock. The valid return values are listed below:

- 0 The registry is locked.
- 1 The registry is temporarily unlocked (it will revert back to locked after a reboot).
- 2 The registry lock is disabled (the RegistryLock value in the registry is set to 0).

```
int UTL_UnlockRegistryNextBoot();
```

This function arms the registry unlock feature so the registry will be temporarily unlocked after the next system reboot.

```
int UTL_SaveRegistry();
```

If the registry is unlocked, this function will flush any registry updates to disk. If the registry is locked, this function has no effect.

```
int UTL_Reboot();
```

This function will shutdown and reboot the system.

AppSrv Configuration Utility (TCU)

TCU is a console-mode utility that can be used to setup basic security and network configuration. It can be run from a command prompt, remotely over the network via telnet, or even over a serial port. It was primarily designed to ease configuration of headless units.

If enabled, TCU will monitor serial port COM1 for a carriage return character (0x0D) for a predefined period of time after boot. If the CR is received, the configuration menu will be displayed. By default, this feature of TCU is disabled on AppSrv. To enable the serial-port feature of TCU, the following DWORD registry values must be set under the key
HKEY_LOCAL_MACHINE\Drivers\Console:

```
EnableTcu = 1 (1 to enable TCU on COM1, 0 to disable)  
ComSpeed = 38400 (baud rate)  
TcuWait = 30 (seconds to monitor port after boot)
```

Launching an Application at Startup - Registry Method

A feature has been added to allow one or more applications to be run when the system is booted. To use this feature, add the application(s) to the following registry key:

```
HKEY_LOCAL_MACHINE\Software\Online Development\LaunchIt
```

The path must be a string value, and must be the full path to the executable file. For example:

```
ExampleApp = \IDE Disk1\Example.exe
```

Launching an Application at Startup - Batch File Method

Applications may also be launched at startup from a batch file. The system will check the CompactFlash (if it exists), then the DiskOnChip for the existence of the appropriately-named batch file. The search order and file names are listed below:

- 1 \Ide Disk1_runfirst.bat
- 2 \DiskOnChip_runfirst.bat
- 3 \Ide Disk1_run.bat
- 4 \DiskOnChip_run.bat

If a file of name `_runfirst.bat` is found, a command shell is started to execute the file. No other batch files are run, and none of the applications under the `LaunchIt` key in the registry are started.

If a file of name `_run.bat` is found, a command shell is started to execute the file. No other batch files are run, but the applications under the `LaunchIt` key in the registry are started.

Known Issues

If power is removed while a file system write to the DiskOnChip is in progress, there is small possibility that the file system can be corrupted. Therefore, it is advised that applications avoid writing to the DOC. If you must write to a file, a CompactFlash card should be used.

Revision History

Version 1.23 12/21/06:

- handles Modbus Node Addressing through a Tcp/Ip to serial gateway.
- allows Modbus commands to be sent using C functions.
- The Modbus Master EZ Widget will allow the user to set up a set of Modbus commands at the Modbus Interface Instance level.
- A number of C functions were modified to handle the Modbus Commands. They require a handle to the Modbus Interface Instance to access the commands.
- Modbus Slave driver functions added

Array support added to the 1769 tag configuration dialog V1.22 1/9/06:

Added SQL Server CE 2.0 component V1.21 12/12/05:

Installed all QFE's through 10/28/05 V1.20 6/1/05:

Installed all QFE's through 5/26/05 V1.19 2/17/05:

Added SNTP service V1.18 6/23/04:

- Added support for up to 256MB RAM
- Installed latest QFE's through 6/16/04
- Added ICS and network bridging components
- 1769 IO API enhanced to support module configuration while scanning, online output inhibit, and online program/run mode

V1.16 2/25/04:

- Corrected problem with embedded IO output 2

Changed EZ BufferRamAllocation back to 0x400000 V1.15 2/24/04:

- Installed latest QFE's through 2/20/04
- Increased number of DHCP retries and decreased retry interval to better handle situations where DHCP server is unavailable at boot.

Updated DP83815 driver to v1.36.3 V1.14 2/6/04:

- Added SOAP toolkit client
- Installed all available QFE's through 2/2/04
- Changed EZ BufferRamAllocation to 0x800000
- Implemented registry lock
- Added Utility API
- Added registry lock support to RegSave
- Added registry lock support to TCU
- Disabled default telnet authentication to allow headless setup

Added backplane buffer support to CPCBPAPI V1.13 12/15/03:

- Installed latest QFE's (through 12/6/03)
- Added XML HTTP component
- Modified TCU to display platform information on startup

Updated EtIplApi to v1.02 V1.12 11/26/03:

- Installed latest QFE's (through 11/20/03)
- Added SRMP component
- Added VPN components
- Updated sdnceapi and changed default IST priority from 160 to 60

Updated cpclio V1.11 10/16/03:

- Installed latest QFE's (through 10/14/03)
- Added Unimodem component
- Added PCL printer driver
- Added Ethernet/IP API

Changed USB hard disk folder name to 'USB Disk' V1.10 9/17/03:

- Migrated to CE 4.2
- Added _run.bat support
- Added Save Registry desktop shortcut

Added background DiskOnChip defragmentation V1.06 8/19/03:

- Added DCOM Remote Access
- Updated Flex API
- Added ActiveSync
- Changed high-temperature handling (does not shut down)

V1.05 6/23/03:

- Added SystemPath registry key (adds \DiskOnChip to default search path for DLL's and Exe's)
- Updated 69 IO API (cpcbapi.dll)

Disabled anonymous FTP access V1.04 6/4/03:

- Added .NET Compact Framework
- Modified cpclio.dll so that EIO will not be accessed unless it is one of our EIO boards (for example, no Eio if Fieldbus)

Installed 1Q2003 CEPB QFEs V1.03 5/1/03:

- Added CSMTTP email library
- Added email configuration/test utility TESTMAIL
- TSM updated to v1.0.1.3
- CPCLIO API and DDI modified to allow multiple processes to access LIO devices
- Ethernet Port 2 (leftmost) default configuration changed from DHCP to static IP address of 192.168.0.1, subnet mask 255.255.255.0

V1.02 3/31/03:

- Installed most recent WinCE 4.1 QFE's
- Added SOTI Pocket Controller Beta server (www.soti.net)
- Added CERDisp component (modified cerdisp.exe to accept host name on command line)
- Added Configuration Utility (tcu.exe) for basic device configuration
- Added HKLM\SOFTWARE\Online Development\EZ1131\RSQ key
- Updated TSM to improve performance
- CPCbpapi updated to v1.02

CPClio updated to v1.02 V1.01 1/22/03:

- Updates to TSM service

Discovery Release 1 V1.00 1/17/03:

API versions bumped to 1.0 V0.21 1/14/03:

- Added Toolhelp feature
- CPCbpapi updated to v0.20
- DOC driver updated to v5.1.4.1 (from M-Systems)

V0.20 1/8/03:

- Updated default EZ registry keys for task priorities
- Added IOCTL_SERIAL_SET_FIFO_LEVEL in serial port driver

Cpcbapi feature added to enhance IO scan performance V0.19 12/11/02:

- Disabled KITL and Target Control
- Disabled serial debugging output

Updated System Maintenance Server V0.18 11/26/02:

- Storage/program memory division setting is now saved in registry and restored on boot
- Updated default EZ registry keys for watchdog and task priorities
- Cpcbapi changed to stop scan on close

Storage manager applet modified to prevent making disk unbootable V0.17 11/15/02:

- Added EZ registry key ContinuousTaskSleepTime
- Allow LIO interrupts during power-fail
- Added index values to registry for COM1 = 3F8, COM2 = 2F8

Added System Maintenance Server V0.16 11/8/02:

- Added Flex API to platform and SDK
- Added USB printer class driver (untested)
- Added support for _runfirst.bat
- Added C++ structured exception handling and RTTI

Disabled COM1 debug output V0.15 10/29/02:

- Updated cpclioggi to fix RTE watchdog timeout
- Added driver for Quad Serial CAM
- Changed default RTE registry keys to point to DOC instead of CompactFlash (LaunchIt and FindIt)

V0.14 10/18/02:

- Updated DP83815 Ethernet driver to latest release from National (v1.0.2)
- Added PS/2 keyboard and mouse support (in addition to USB HID - either/both may be used)
- Registry stored on DOC
- Updated BIOS setup utility
- Integrated sdnceapi and cpcbapi into platform build

Corrected problem with DDI CPCIdi_Open94Io function V0.13 10/4/02:

Added driver support for Flex IO and Embedded IO V0.12 9/24/02:

Updated default thread priorities V0.11 9/20/02:

- Optimized ATAPI driver to increase CF performance
- Enabled FATFS cache (1024 sectors)
- Modified the USB HID driver for the MS mouse

Changed EZ1131\ BufferRamAllocation to 0x400000 V0.10 9/18/02:

- First release based on CE 4.1
- Replaced Pocket IE with IE5.5
- Added FTP Server

Provided as two versions for registry on CF or DOC V0.05 8/21/02:

- Added powerfail/reset support
- Default thread quantum = 10ms
- Added temperature monitor
- Added telnet server

- Added EZ registry keys

V0.04 7/17/02

- Added DeviceNet CAM driver
- Disabled demand paging
- First distributed release

V0.03 7/1/02

1.2 Important User Information

Because of the variety of uses for the products described in this publication, those responsible for the application and use of this control equipment must satisfy themselves that all necessary steps have been taken to assure that each application and use meets all performance and safety requirements, including any applicable laws, regulations, codes and standards.

1.2.1 *Preventing Damage from Electrostatic Discharge*

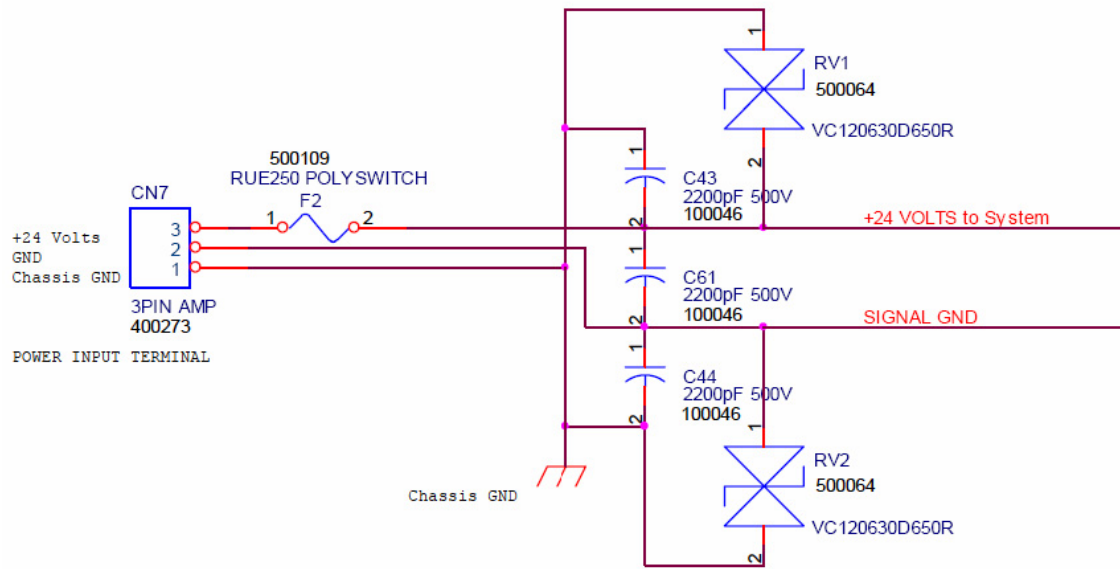
ATTENTION: Electrostatic discharge can damage internal components if you touch them or connector pins. Use these guidelines when handling the module:

- Touch a grounded object to discharge static potential
- Wear an approved anti-static wrist strap
- Do not touch conducting surfaces or connector pins
- Do not touch circuit components inside the module
- If available, use a static-safe work station
- When not in use, keep module in its static-shield box

1.2.2 *Surge Protection for AppSrvCE Power Input*

The AppSrvCE is designed to meet the requirements of EN61000-4-5:1995

The power input schematic diagram is shown below. Transient voltage suppressors RV1 and RV2 are designed to protect the input from surge. However, for the suppressors to be effective, it is imperative that a good chassis ground connection be made on the power input terminal as well as the chassis connection.



1.2.3 Wiring Requirements

- wiring requirements
- input voltage ranges, and output voltage ranges

Wire Type	Wire Size(1)	Wiring Torque
Solid	Cu-90°C (194°F) #14 to #22 AWG	1.13 Nm (10 in-lb) rated
Stranded	Cu-90°C (194°F) #14 to #22 AWG	1.3 Nm (12 in-lb) maximum

(1) Two wires maximum per terminal screw.

ATTENTION: Be careful when stripping wires. Wire fragments that fall into the controller could cause damage. Once wiring is complete, be sure the base unit is free of all metal.

Wiring Recommendation



Before you install and wire any device, disconnect power to the controller system.



ATTENTION

Calculate the maximum possible current in each power and common wire. Observe all electrical codes dictating the maximum current allowable for each wire size. Current above the maximum ratings may cause wiring to overheat, which can cause damage.

United States Only: If the controller is installed within a potentially hazardous environment, all wiring must comply with the requirements stated in the National Electrical Code 501-4 (b).

- Allow for at least 50 mm. (2 in.) between I/O wiring ducts or terminal strips and the controller.
- Route incoming power to the controller by a path separate from the device wiring. Where paths must cross, their intersection should be perpendicular.

NOTE Do not run signal or communications wiring and power wiring in the same conduit. Wires with different signal characteristics should be routed by separate paths.

- Separate wiring by signal type. Bundle wiring with similar electrical characteristics together.
- Separate input wiring from output wiring.
- Label wiring to all devices in the system. Use tape, shrink-tubing, or other dependable means for labeling purposes. In addition to labeling, use colored insulation to identify wiring based on signal characteristics. For example, you may use blue for dc wiring and red for ac wiring.

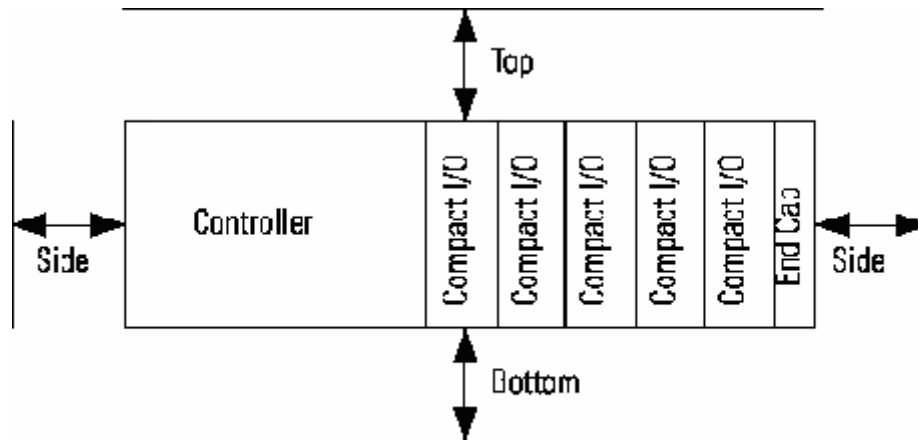
Miswiring Power Connector

The following table shows miswiring conditions and the consequences of improper wiring:

Condition	Result
Operating with Voltage Less than 10.0V dc	This will not damage the base unit. The base unit may not power up. This is not recommended. You must verify that the voltage remains within specified limits.
Reverse Wiring of the Line Terminals (0 to 30V dc)	Reverse wiring will not damage the base unit. The base unit will not power up.
Applied Voltage Level Exceeds the Published Recommended Value (27 VDC)	Exceeding the published recommended voltage may result in permanent damage to the base unit.

1.2.4 Controller Spacing

The base unit is designed to be mounted horizontally, with the Compact™ expansion I/O extending to the right of the base unit. Allow 50 mm (2 in.) minimum of space on all sides for adequate ventilation, as shown below.



1.2.5 **Grounding the Controller**

In solid state control systems, grounding wire routing helps limit the effects of noise due to electromagnetic interference (EMI). Run the ground connection from the ground screw of the base unit to the electrical panel's ground bus prior to connecting any devices. Use AWG#14 wire.

This product is intended to be mounted to a well grounded mounting surface such as a metal panel. The grounding stamping must be connected for DIN Rail mount or panel mount installation.

Refer to the ANSI/IEEE Standard 142–1982 for more information.

The IEEE Recommended Practice for Grounding of Industrial & Commercial Power Systems is an authoritative and informative book on the subject of grounding. It sets forth in a clear and concise manner the problems and solutions to problems concerned with grounding. It also gives the electrical engineer the basis for applying foundations to practical solutions.

Published by:

The Institute of Electrical and Electronics Engineers, Inc.
345 East 47th Street
New York, NY 10017

Contact:

IEEE Service Center
445 Hoes Lane, P.O. Box 1331
Piscataway, NJ 08855-1331

1.3 Environmental Specifications

Temperature:	
Operating	0° to 60°C (32° to 140°F)
Storage	-40° to 85°C (-40° to 185°F)
Mounting: DIN-rail or panel mountable	
Humidity: 5% to 95% non-condensing	
Vibration: 10 to 500 Hz 2.0 G max. peak acceleration 0.012 in (peak-to-peak) displacement	
Shock:	
Operating	30 G peak for 11ms
Storage	50 G peak for 11ms
Weight: 1.7 lbs. (771.107 g)	
Power: 10 to 24 VDC @ 0.25A 27 VDC max. User supplied.	
Battery: Rechargeable Lithium Maintain 30 day charge	

2 Installing AppSrvCE

In This Chapter

- Unpacking the Base Unit..... 23
- Viewing the Base Unit..... 24
- Inserting/Removing a CAM 24
- Mounting the Unit..... 25
- Power Supply Requirements..... 31
- Wiring the Base Unit Power Supply 32

This chapter will guide you through installing and booting the system. For the installation, you will need a base unit, power supplies, and the installation instructions that shipped with each.

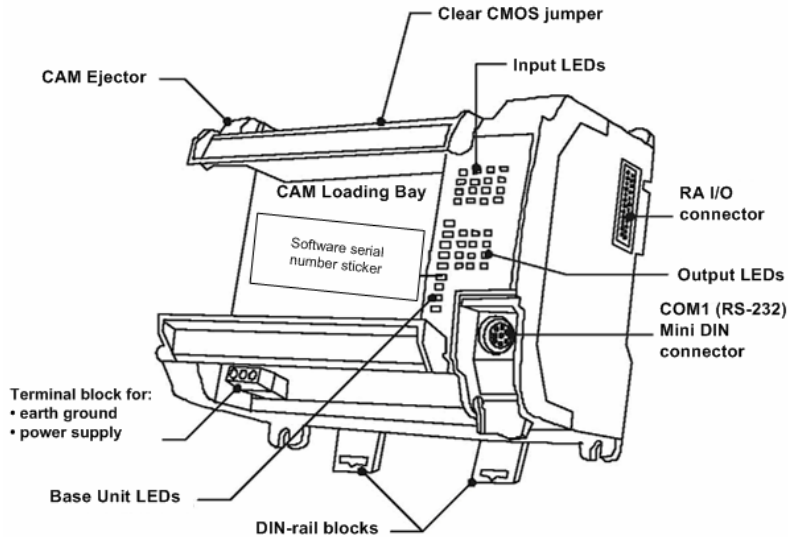
2.1 Unpacking the Base Unit

The base unit is shipped with the following components:

- serial adapter cable
- installation instructions
- software license envelope containing Getting Started guide

Important: The base unit ships with the EZ Design Studio runtime engine loaded on its disk-on-chip.

2.2 Viewing the Base Unit



Locate the following on the Base Unit:

- CAM ejector lever
- LEDs supported on units with AppSrvCE Embedded I/O
- RA I/O connector to Compact or Flex I/O modules on the DIN-rail
- COM1 port (RS-232 isolated programming port)
- DIN-rail locks
- Terminal block for wiring power supply
- LEDs to report base-unit and customer-defined status
- Bay for inserting/removing a CAM (includes software serial number label)

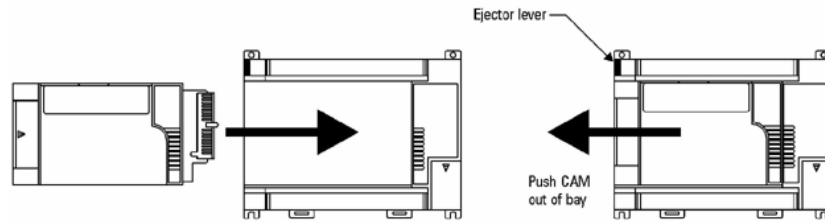
2.3 Inserting/Removing a CAM

You must insert a CAM into the base unit for it to operate.

ATTENTION: Remove power before inserting/removing a CAM. Otherwise, you risk damaging the electronics in the CAM or base unit. If your system is controlling a machine or process, physical damage or personal injury could result.

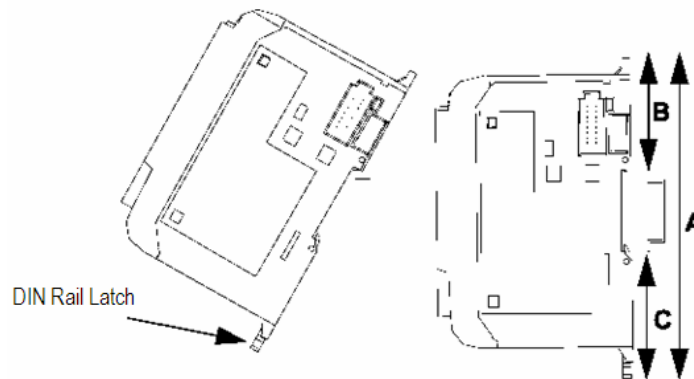
To insert a CAM, place it in the loading bay and slide it to the right until its connector mates with that of the base unit. Push firmly to seat the connector pins.

To remove a CAM, pull the CAM ejector lever on the base unit while simultaneously pushing the CAM out of the bay (toward the left).



2.4 Mounting the Unit

The base unit and expansion I/O DIN rail latches lock in the open position so that an entire system can be easily attached to or removed from the DIN rail. The maximum extension of the latch is 15 mm (0.67 in.) in the open position. A flat-blade screw driver is required for removal of the base unit. The base can be mounted to EN50022-35x7.5 or EN50022-35x15 DIN rails. DIN rail mounting dimensions are shown below.

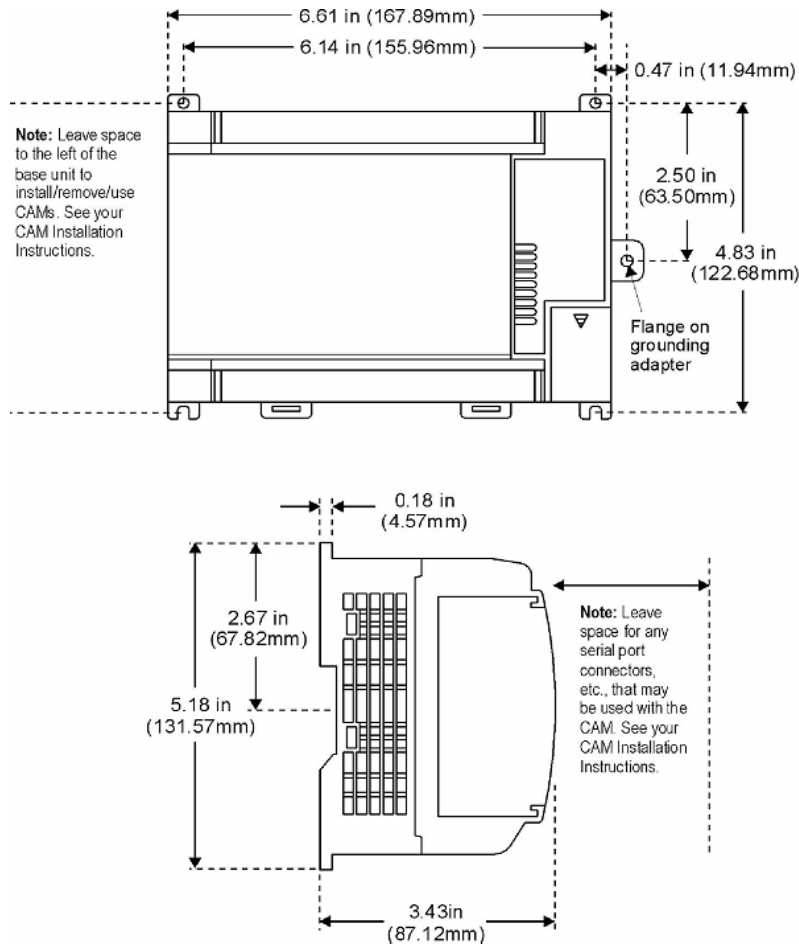


Dimension	Height
A	DIN latch open: 138 mm (5.43 in.), DIN latch closed: 118 mm (4.65 in.)
B	47.6 mm (1.875 in.)
C	47.6 mm (1.875 in) DIN latch closed 54.7 mm (2.16 in.) DIN latch open

ATTENTION: The base unit must be mounted horizontally on a vertical plane in a grounded conductive enclosure. Any connections that exit the enclosure must exit through a bulkhead connector mounted to the frame of the conductive enclosure. The base unit ships with a grounding adapter attached to the back of it.

Important: To protect the base unit from debris during installation, use the debris cover that shipped with it. Wrap cover around unit, covering top and bottom ventilation slots. Remove cover after installation is complete.

Important: The dimensions below are for a base unit with a blank CAM. When mounting your system, be sure to leave adequate space for the insertion/removal of CAMs, PC cards, serial port connectors, power supplies, I/O, etc.

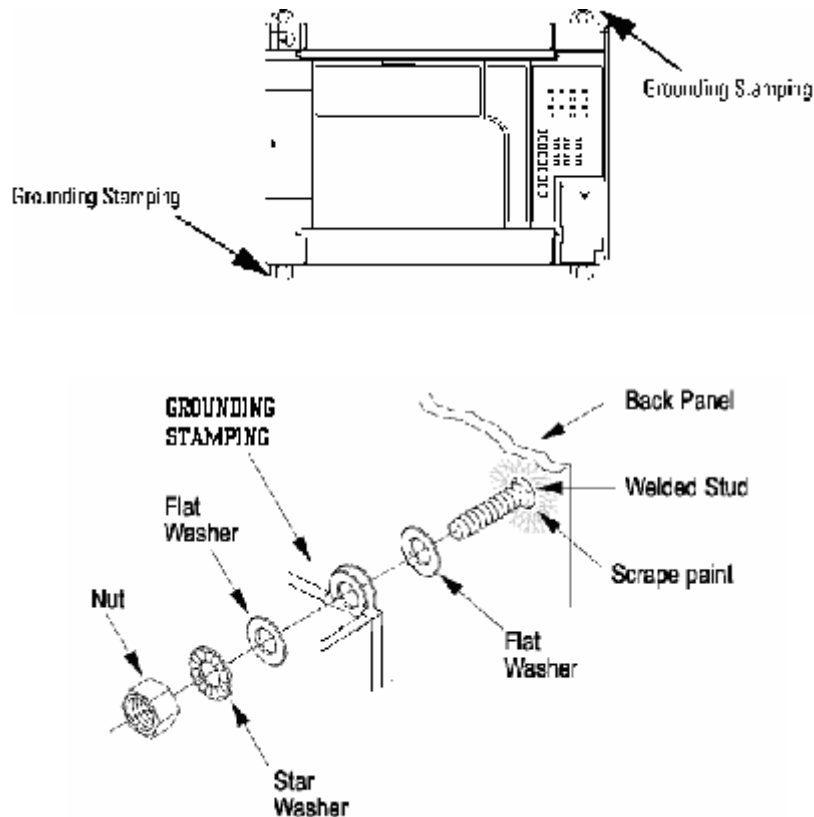


2.4.1 Panel Mount

Mount to a well grounded mounting surface such as a metal panel. (Make sure that the placement of the base unit meets the recommended spacing requirements).

Note: It is recommended to use all four mounting positions for panel mounting installation. Mount to panel using #8 or M4 screws to connect the grounding stampings to a well grounded mounting surface such as a metal panel.

Stud Mounting to Panel



- 1 Retain the grounding adapter, which provides an electrical ground connection for the system.
- 2 Drill five mounting holes.
- 3 Using appropriate hardware, mount the base unit.

2.4.2 *DIN-rail*

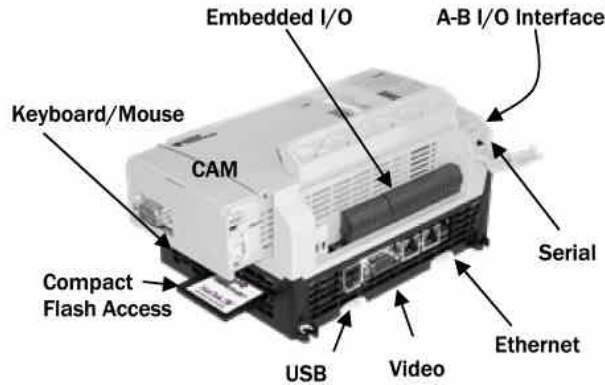
If you are using the base unit with Flex I/O modules to interface with machine inputs and outputs, mount the base unit to DIN-rail. Use an end anchor on each end of the system to secure it.

To install the base unit onto the DIN-rail:

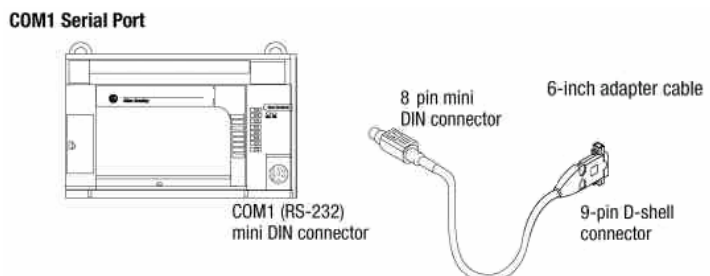
- 1 Mount your DIN rail to a well grounded mounting surface such as a metal panel. (Make sure that the placement of the base unit on the DIN rail meets the recommended spacing requirements).
- 2 Remove the grounding adapter
- 3 Pull down the two rail locks on the back-bottom of the base unit.
- 4 Hook the base unit onto the upper flange of the DIN-rail. Then push the base unit onto the lower flange of the DIN-rail.
- 5 Push up the rail locks until they snap into place.

- Slide the base unit to its intended location on the DIN-rail and secure it with an end anchor on the left-hand side. Use the other end anchor to secure the base unit, or if using RA I/O, the right-most module.

Location of the Connections on the Base Unit Using the Serial Port on the Base Unit



The COM1 serial port on the base unit supports RS-232 communication at baud rates up to 115.2K bits/sec. The base unit includes an adapter cable with a standard 9-pin D-shell connector to connect to your hardware.

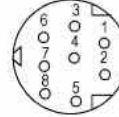


Important: To connect to a terminal device, such as a PC, using the COM1 serial port, you must attach a 9-pin null modem serial cable to the adapter cable. Total cable length should not exceed 32 feet (10 meters).

The following table lists the pin assignments for the adapter cable:

Mini-DIN Connector		9-pin D-sub Connector	
Pin	Signal	Pin	Signal
1	DSR	1	DCD
2	DTR	2	RX
3	RTS	3	TX
4	RX	4	DTR
5	DCD	5	common
6	CTS	6	DSR
7	TX	7	RTS
8	Common	8	CTS
-	-	9	not used

Pin layout on the base unit
(Mini DIN female)

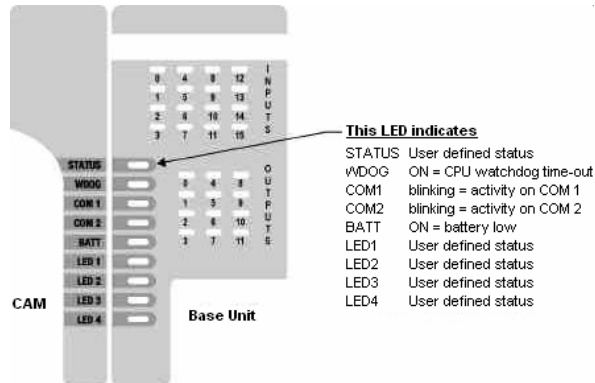


Pin layout on the 9-pin D-sub
cable connector (male)

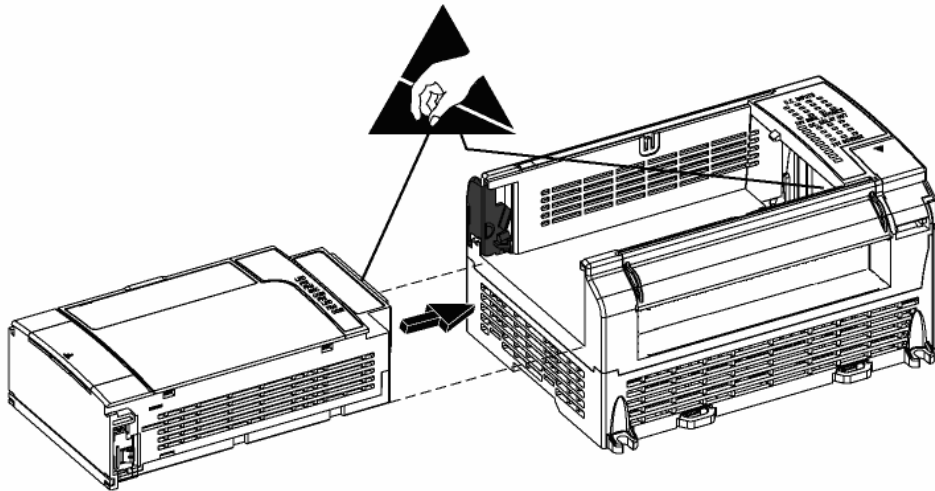


Using LED Indicators for Status of Base Unit

Locate the single column of Base Unit status LED indicators on the base unit. The LED labels are located on the CAM:



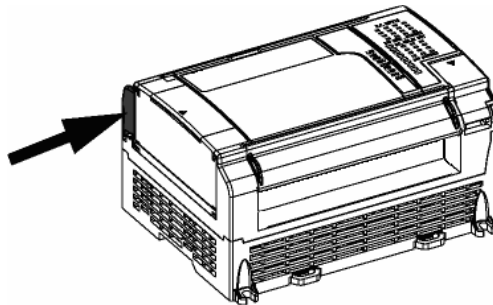
2.4.3 *Installing the CAM*



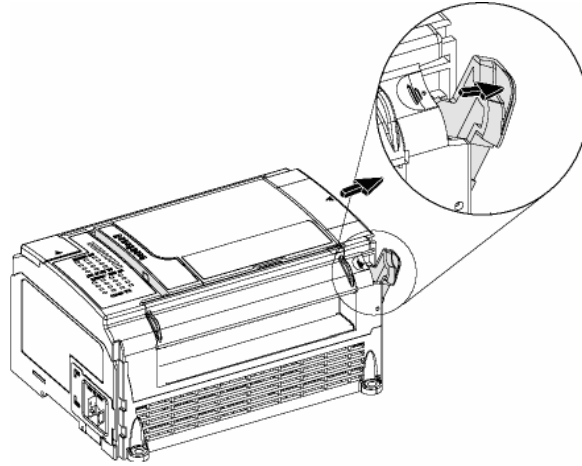
- 1 Be sure base unit power is off.
- 2 Slide the CAM into the base unit using the guide rails for alignment.
- 3 Push until a click is heard.

Important: It is critical that the CAM is fully engaged and locked into place

- 4 Make sure the actuator is pushed closed.



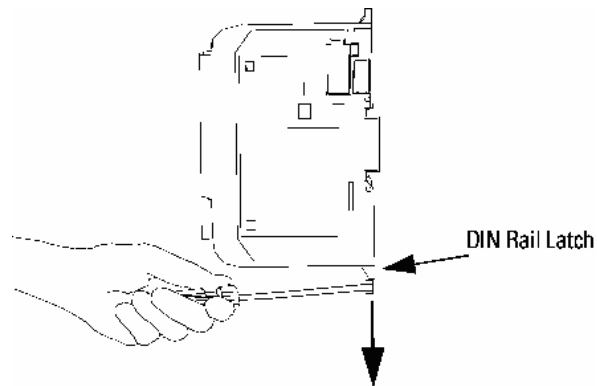
- 5 To remove the CAM from the base unit, make sure base unit power is off. Push the actuator to the open position until the cam is ejected slightly. Once the CAM has been ejected, it can be removed from the base unit.



2.4.4 Removing the Base Unit from the DIN Rail

To remove your base unit from the DIN rail:

- 1 Place a flat-blade screwdriver in the DIN rail latch at the bottom of the base unit.
- 2 Holding the base unit, pry downward on the latch until the latch locks in the open position. Repeat this procedure with the second latch. This releases the base unit from the DIN rail.



2.5 Power Supply Requirements

We recommend the Rockwell Automation power supply (RA catalog number 1794PS1) or any Class-2 power supply. Refer to the following table to determine the input voltage for your system:

Base Unit Nominal Voltage	Input Voltage Range	DC/DC Isolation
24 V dc	10 to 24 V dc	No

ATTENTION: Maximum current draw through the system 24W (0.6 A @ 24Vdc or 1.3A @ 12Vdc, 27Vdc max.). You must provide additional power supplies as needed for your embedded I/O and Compact or Flex I/O modules.

2.5.1 For Rockwell Automation Flex I/O

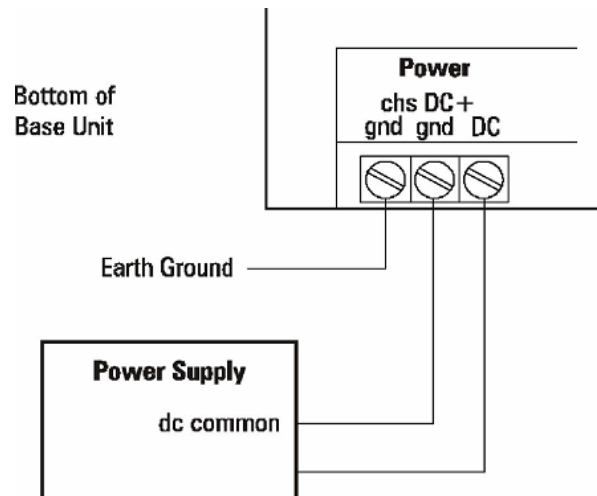
The base unit provides 5V dc to power the logic side of Flex I/O. You must apply power directly to Flex I/O modules to meet their external power requirements. See RA Flex I/O datasheets.

2.5.2 For Rockwell Automation Compact I/O

The base unit provides 5V dc to power the logic side of the I/O. You must apply power directly to Compact I/O modules to meet their external power requirements and any 24V modules. See RA Compact I/O datasheets.

2.6 Wiring the Base Unit Power Supply

Locate the two 3-wire terminal blocks. Use them to connect a power supply for the base unit.



The AppSrv is designed to meet the requirements of EN61000-4-5:1995. The AppSrv contains noise filtering and transient voltage suppression circuitry on the input power connection. For this circuitry to be effective, it is imperative that the chassis ground terminal of the input power connector be connected to chassis (earth) ground via a heavy gauge ground strap.

3 EZ Design Studio

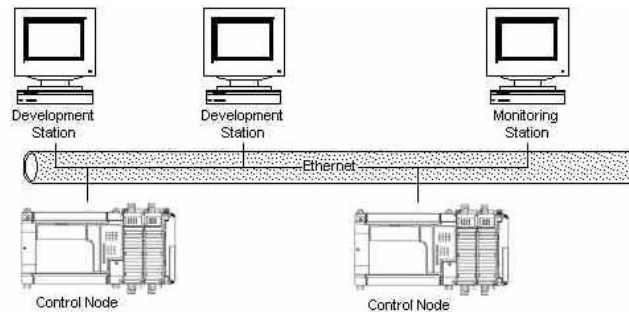
In This Chapter

- What is EZ Design Studio? 33
- Flowchart 34
- EZ Design Studio Software Components 34
- Installing EZ Design Studio 37
- EZ Design Studio Applications 38

Note: The procedures in the following section require that you have successfully installed the development system, that compatible physical control node hardware is properly installed, and, that if you plan to use network communications, that the development system and control node are networked. If not, refer to the Installation Instructions.

3.1 What is EZ Design Studio?

EZ Design Studio automation software is a scalable development system and real-time control engine that runs re-usable program application code on the 7000-ADM controller.



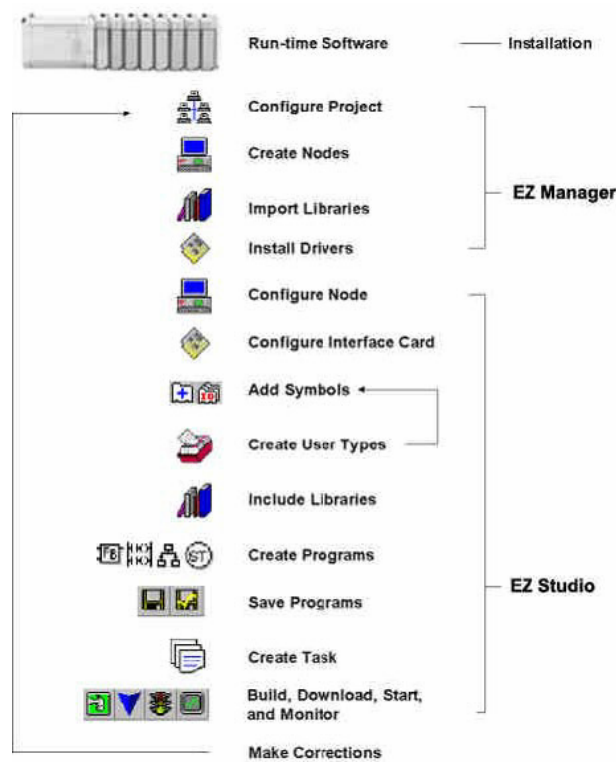
The distributed architecture offers a central database for program development, debugging and monitoring which provides security and allows users backup capability.

EZ Design Studio real-time control engines are programmed using a full set of re-usable objects including network nodes, interface cards, and program control units, which include standard, user-defined, and 'C' function blocks. Use the local I/O on the 7000-ADM or connect Rockwell Automation 1769, 1794, or 1793 I/O

directly to the controller. Device network communications are also available through Modbus, Modbus TCP/IP and other protocols.

EZ Design Studio development software consists of configuration and programming editors that run on the development station, and a real-time control engine that runs on the control node platform. The control node executes the control logic and interfaces to the I/O system. The development station and control node communicate over an Ethernet network. Multiple workstations and control nodes can reside on the same Ethernet network and communicate with each other.

3.2 Flowchart



3.3 EZ Design Studio Software Components

EZ Manager – EZ Manager provides a project-level view of your control system. You can look at and monitor any control node on the network. You can also add and delete control nodes, libraries, and device drivers.

EZ Design Studio – EZ Design Studio lets you develop application programs that run within a control node.

Libraries – Contain function and function block algorithms.

Device Drivers – Allow the runtime engine to communicate with external devices.

Portable Application – The code generated as a result of an EZ Design Studio build and **Code** download to a control node. It is independent of the control node platform.

Run Time Engine – The program that runs your portable application code. A different runtime engine is available for each control node platform.

Operating System – The operating system used by the control node platform.

EZ Monitor – EZ Monitor is an EZ Design Studio utility that lets you monitor control node activity over the network. You can run EZ Monitor independently of other EZ Design Studio software.

Watch Window – Watch Window lets you monitor, set, and force program variables on the control node. It runs within EZ Design Studio.

Human-Machine Application software – separate from EZ Design Studio that typically Interface displays and controls control node operation in a sophisticated manner. HMI's can run locally on the control node, on the development station, or on a monitor station.

Connectivity Export/import data values to/from an external database.

Source – The collection of control objects in an EZ Design Studio control node. This collection of control objects is built and downloaded as portable application code. You can also download the source code, which permits it to be uploaded from the control node to another development station, for example.

Debug Database – Translates symbol names to their memory addresses within the control node. It allows the watch window or an HMI for example to refer to a symbol by name.

Toolkits – Additionally, toolkits provide the capability to write user-defined functions and function blocks, and I/O drivers, and to connect to the Symbol Server to access symbol data.

Each development station can maintain one or more control nodes based on TCP/IP address. As well, a single control node could be maintained by more than one development stations (with appropriate co-ordination). Monitoring stations can run development station software or a subset of it, or HMI software in order to monitor and/or control the application. Control nodes can also communicate with each other in order to distribute complex control applications over multiple control nodes.

3.3.1 ***EZ Design Studio Features***

EZ Design Studio features include:

- Windows Tree-structured EZ Manager to view and edit control NT/2000/XP nodes;
- Multiple document interface (MDI)

Development Environment

- EZ Studio to configure and program control applications.
- IEC Languages and Standard Function Library
- User-Defined Functions
- Re-useable Program Objects
- Portable Application Code
- Scalable Controllers
- Task Objects and Task Management
- Use any combination of the five IEC 61131-3 programming languages in one control application:
 - Sequential Function Chart for machine sequencing control and for control applications that have multiple operating modes, such as manual mode and automatic mode.
 - Ladder Diagram (Relay Ladder Logic) for discrete logic programs.
 - Function Block Diagram for analog signal processing and any control process or algorithm that runs on a continuous basis.
 - Structured Text for complex algorithms, string and file operations, and manipulation of data structures best done in a procedural (text based) language.
 - Instruction List for compact assembly language-like programming.
 - Function Library includes all defined IEC 61131-3 functions and function blocks and is augmented with many custom developed functions.
- Create your own functions and function block libraries in 'C' and import them into EZ Design Studio to further enhance functionality.
- Re-useable programs and configurations. Central database for storage and management of all program objects. Node Templates and Object Archive to easily duplicate and reuse all or parts of existing programs and configurations. Libraries simplify re-using functions and function blocks.
- Run the same application code regardless of software platform: Windows XP, Windows CE.
- A minimum hardware platform is required to run the Portable Application Code. You can scale the hardware features to meet the needs of the application by upgrading with a faster processor, additional program memory, flash memory (only Industrial Grade Compact Flash should be used), retentive memory, and other components.
- Place a program object in one or more tasks and have it run continuously, periodically at the task scan rate, or periodically at the I/O scan rate.
- Networking and Remote multi-node programming. Each controller is a node
- Distributed Control on the network. You can easily distribute control applications among nodes to control loading and have control at the point of operation. Controllers communicate with each other over the network.
- On-Line Monitoring. Observe control application program operation graphically and in real-time, within the same editors used to create the control application programs.
- Integrated Watch Window to watch and force variables; Control Node
- Monitor to observe node status (running, faults, statistics). Monitoring and HMI can run on the development station, the monitor station, or locally at the embedded node.
- Online Help Tutorials and sample applications are provided to assist System you in quickly learning to use EZ Design Studio.

3.3.2 **Platform Requirements**

The following lists minimum recommended requirements.

Development Station

Hardware

- Minimum: 133MHz with 64Mb of RAM.
- Recommended: 500MHz with 128Mb of RAM.
- 60 MB free disk space for installation, plus space for application development.

Software

- Windows NT/2000/XP.
- Microsoft Internet Explorer 5.0 or later to view help topics (help viewer can be installed from product CDROM).
- Microsoft Embedded Visual C++ 6.0 or later for C Function Block Toolkit support.

Controller

AppSrvCE

ProSoft Technology controller with Windows CE .NET. Includes retentive memory, on board I/O, Ethernet, and status LEDs.

EZ-RTE Simulator for Windows NT/2000/XP

Execution engine that runs on a Windows 2000/XP workstation and can be used for application simulation and debugging.

3.4 **Installing EZ Design Studio**

Insert the product CD and a menu dialog will open. Click the **Install EZ Design Studio** button and follow the instructions during the installation to install the Workstation software.

Note: To successfully install EZ Design Studio, you must have administrator privileges for the domain in which you are logged in.

3.4.1 **Post-Installation**

- To start EZ Design Studio development, go to the Windows Start menu and select: Programs | EZ Design Studio | EZ Manager or Programs | EZ Design Studio | EZ Design Studio. Refer to the EZ Manager or EZ Design Studio help for more information.
- Refer to the instructions from the EZ Design Studio Help menu for information on using your runtime platform.

3.4.2 Runtime Connection

You can connect to EZ Design Studio Runtime Engine running on the AppSrvCE disk on chip via an Ethernet connection in one of the following ways:

- Over an Ethernet network.
- Through a network hub.
- Direct network card to network card connection using a crossover cable (peer-to-peer networking).

To make a crossover cable, wire it as follows. The wire colors are an example and may not be specific to your cable. You can also find information on the Internet on making cables and adapters.

Wire	Connector 1 - Pin	Connector 2 - Pin
Orange Stripe	1	3
Orange Solid	2	6
Green Stripe	3	1
Green Solid	6	2

3.4.3 Where to Get More Information

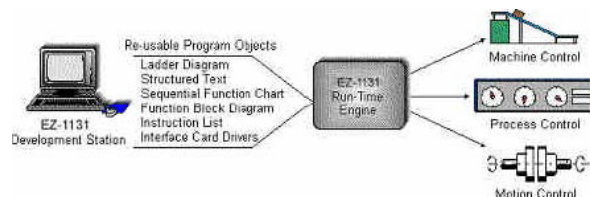
EZ Design Studio has a complete online help system. Click the Help button or press F1 to view online help for all the configuration and programming editors. The help files also include a comprehensive Language Reference.

3.5 EZ Design Studio Applications

EZ Design Studio software is ideally suited for a wide variety of machine and process control applications. These include embedded control applications requiring real-time diskless operation, plant-wide distributed machine applications, and integrated workstation applications.

EZ Design Studio is easily programmed for any type of control application: discrete control or process control. It features user-friendly graphical configuration and program editors that support the IEC 61131-3 programming languages. Networking features allow the development station to communicate with one or more runtime engines (control nodes).

The image below shows a high-level overview of the programmable control system. Hardware components in the control system consist of a development station, the runtime engine controller, and the I/O devices. Major software components include the development software used to configure the re-usable program objects (that are compiled into the application code), the runtime engine, and the application code executed by the runtime engine.



4 Application Development Overview

In This Chapter

- Configuring and Programming 39
- Running and Operating 39
- Procedures 40

A development station is used for configuration and programming the control application, which is then downloaded to the control node where control logic is executed. The development station can be used to monitor and control node operation to view, set, and force control variables (for example, I/O point values) in the control logic. Control monitoring and local HMI are also possible on the control node using other application software.

4.1 Configuring and Programming

Configuration refers to adding control nodes, interface cards, tasks, symbols, and program control units to a project and setting their properties. A task is a container for program control units and can be considered the executable for the control logic. Symbols are variables used within the control logic (for example, program variables and I/O points). Program units make up pieces of the control logic; they must be included in a task to be run. Programming refers to developing the control logic program control units, which are the algorithms or pieces of code that comprise the control strategy. Program control unit development is supported using the Sequential Function Chart (SFC), Relay Ladder Logic (RLL), Structured Text (ST), Instruction List (IL), and Function Block Diagram (FBD) languages. Editors for each type of configuration and programming language simplify the configuration and programming functions.

4.2 Running and Operating

After a control node application is configured and programming is completed, the control application components are assembled (the build process) and downloaded to the control node. The development station (or monitoring station) controls execution of the control logic by starting and stopping the control node. The Control Node Monitor monitors control node operation and the Watch Window displays, sets, and forces control variable values.

4.3 Procedures

4.3.1 *Embedded (Physical Control Node)*

Procedure	What it does
Runtime Software	<p>Prepares the physical control node to accept communications, including downloading of programs, from the development system. If you do this before configuring the virtual node in the EZ Design Studio, it can notify you that communications have been successfully established at that time.</p> <p>The AppSrvCE comes pre-installed with the EZ RTE, but the EZ RTE must be running on the AppSrvCE to communicate with the development system.</p>

4.3.2 *EZ Manager*

Procedure	What it Does
Create Control Nodes	<p>Adds a new control node to the project. You must do this before you can edit the control node in EZ Design Studio. Refer to EZ Manager help.</p>
Import Libraries	<p>Imports external function and function block libraries into the project. Libraries supplement and extend the capabilities provided by the standard library. You must import a library before you can include it in a control node configuration. Libraries can be installed during product installation. Additional libraries may become available after product release and these can be installed from the EZ Manager. Refer to EZ Manager help.</p>
Install Drivers	<p>Drivers are used to communicate with interface cards and other devices. Install drivers using EZ Design Studio's Driver Install Utility. Refer to EZ Manager help.</p>

4.3.3 *EZ Studio*

Procedure	What it Does
Configure Node	<p>Establishes communication and memory parameters of the physical node. You must configure the node before you can communicate with it. After you have configured the node, you can verify that it is on-line. Refer to Node Configuration in the EZ Design Studio help.</p>
Configure Interface Card	<p>To do any input-output, you must be able to communicate with the local I/O, an interface card or driver. Each interface card or driver has its own configuration utility. Typically tags are created when you configure a card to which you must then assign symbol names. Refer to Interface Card Editor in the EZ Design Studio help.</p>
Add Symbols	<p>For each I/O tag you wish to read or write you must assign an I/O symbol. Additionally, you may have information which you want to share between programs, you do this using Global symbols. Refer to Symbol Operations in the EZ Design Studio help.</p>

Procedure	What it Does
Create User Types	Often it is convenient to pass data as a structure rather than individual symbols. To do this, you must first create a user type structure and then assign a symbol name to it. Refer to User Structures in the EZ Design Studio help.
Include Libraries	This is optional. If you have external libraries (containing function and function blocks), you must include them in the node after they have been imported into the project using EZ Manager's Object Archive utility. Some interface cards have libraries. You must include the libraries to gain full access to the features of the driver. Refer to Library Operations in the EZ Design Studio help.
Create Programs and Local Symbols	Program in any of the five IEC 61131-3 languages to control logic and perform operations. Refer to Function Block Diagram Editor, Relay Ladder Logic Editor, SFC Editor, and Structured Text Editor in the EZ Design Studio User Guide for information on using the editors. Refer to the Language Reference for general programming information, function and function block descriptions, and program operation.
Save Programs	Look for the asterisk (*) in the title bar. If you do not save your program changes, they will not be included in the build! You can simply save the program or save and validate it. Save and validate checks the structured text of the program and displays errors if any. Go back and correct any program errors if needed. If you successfully save and validate a program, you will not encounter program compile errors during a build. Refer to File Operations in the EZ Design Studio help.
Create Tasks	Tasks determine when and how programs are run. A program must be in a task for it to run at all. More than one program can be in the same task, and more than one task can be in a control node. Refer to Task Operations in the EZ Design Studio help.
Build	This assembles all the control system components into the structured text that is executed by the runtime engine in the control node. Compiler, linker, and binder errors are caught and displayed in the output window. Refer to On-Line Operations in the EZ Design Studio help.
Download	Downloads the executable control strategy to the runtime engine. Refer to On-Line Operations in the EZ Design Studio help.
Start	Starts the control application. Refer to On-Line Operations in the EZ Design Studio help.
Monitor	Monitors program operation (SFC, RLL, FBD) and allows you to see program flow or data values, depending on the program control unit type. Refer to Monitoring Program Operation in the EZ Design Studio help. You can also use the Watch Window to view, set, or force data values in any program control unit, and the EZ Node Monitor (or EZ Manager) to monitor node and task status and fault status. Refer to Watch Window Operations in the EZ Design Studio help.

Procedure	What it Does
Make Corrections	After observing control system operation, if there is incorrect behavior, go back and make any corrections as needed. If changes are made, at minimum, you must repeat the build, download, start, and monitor steps.

5 Running EZ Design Studio

In This Chapter

- Planning 44
- Startup and System Configuration 44
- Logic Control Programs 45
- Scheduling Program Execution 45
- Building and Downloading 45
- Running and Monitoring 46
- Using the Watch Window 46

EZ Design Studio has two main components: EZ Manager and EZ Design Studio. EZ Manager provides a project-level view of your control system. You can look at and monitor any control node on the network. You can also add and delete control nodes, libraries, and device drivers. EZ Design Studio lets you develop application programs that run within the control nodes.

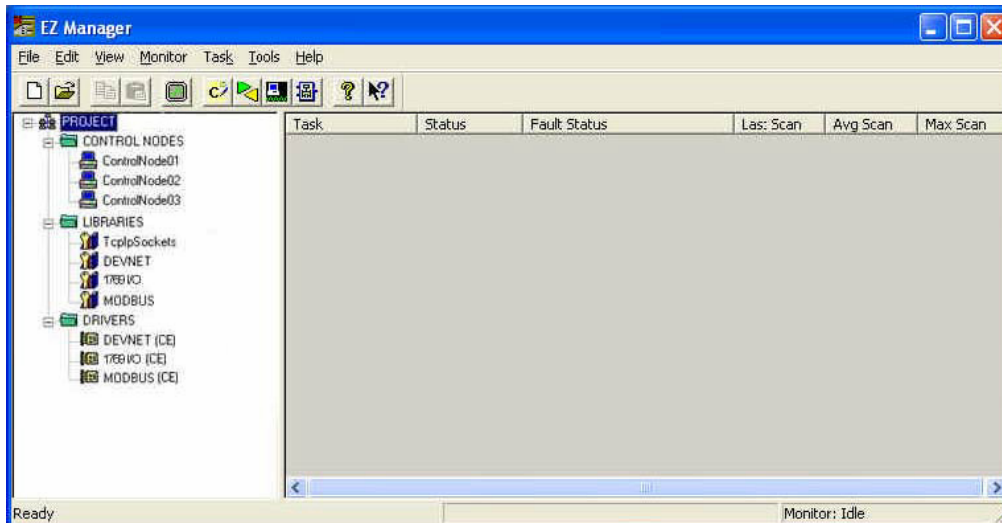
To start EZ Design Studio



Locate the EZ Design Studio menu on the Windows Program menu and click on EZ Manager.

Initially, you will not see any control nodes in the object tree, but if you have installed drivers and libraries, you will see them under their respective folder.

The object tree lets you add and remove control objects from the project. You can also open control nodes by double-clicking on one. The right-hand portion of the EZ Manager screen is used to monitor control nodes. Refer to the EZ Manager help and the EZ Design Studio User Guide for more information. The Where do I Start?, Quick Start, and Building Your First EZ Design Studio Application Tutorial sections in this manual provide an introduction to using EZ Design Studio.



5.1 Planning

The EZ Design Studio system architecture consists of a Windows Development/Monitoring Station that contains the application program editors and utilities and communicates with one or more Runtime Control Nodes that execute the application programs. The first step is to build control logic to meet the application requirements in the most effective manner. After the control logic is developed and the logic control applications have been built, the control logic is downloaded to the Control Node(s) to be executed, monitored and tested.

To do this, first review the specification of the application. Review all the functions to be performed in the application. Determine the number of control nodes required and the functions to be performed at each control node. Determine the number and types of I/O points needed.

5.2 Startup and System Configuration

Start the EZ Design Studio Development System. EZ Manager and EZ Design Studio are used to create and configure each Control Node for the project. Make sure each Control Node on your network has a unique TCP/IP address. Install the software drivers needed for your I/O hardware with the Driver Installation Utility. Configure each I/O card with the Interface Card Configurator and use the

Symbol Editor to assign symbolic names to all of the I/O points in the system. The symbolic names will be used within the application programs.

5.3 Logic Control Programs

For each function to be performed in the application, select which application language is best to use for that function: RLL, SFC, FBD, ST or IL. Relay Ladder Logic (RLL) diagrams are best for discrete logic. Sequential Function Charts (SFC) are best for machine sequencing control and modal operation (applications that have more than one operating mode, for example manual and automatic). Function Block Diagrams are best for continuous process control and control algorithms that run on a continuous basis. Structured Text (ST) is good for complex algorithms, string operations, file operations and manipulation of data structures best done in a procedural (text based) language. You can combine several of the IEC 61131-3 languages together in the same application program, such as RLL programs inside of SFC action blocks. So you can choose the most efficient language to use for any circumstance.

As you are entering your application programs, you can use the Symbol Editor to add new local (visible to only one program) and global (visible to all programs within a control node) variables as needed. Optionally, you can create your own functions or Function Blocks in 'C' or other IEC 61131-3 languages, if necessary, and incorporate them into the project. Add one or more application programs from the EZ Studio program list into the program units list in the task editor.

Do not forget to add appropriate fault handling to your project to manage things should something go wrong. In EZ Design Studio, two special program control units can be created to assist you: a startup program that runs automatically whenever a task is started; and a fault routine that runs only when a major fault occurs.

5.4 Scheduling Program Execution

For each of the application programs created above, you must decide how frequently the program should be executed. The Task Editor is used to create and configure task scheduling units (into which the application programs are placed) that control when the programs are executed. Multiple tasks can be created that run either periodically with different priority levels that you select, or continuously in the background as time allows. In this way you can schedule your task execution to make optimal use of your Control Node hardware. To monitor application program execution in real-time, double click on the program name in the task editor and select the monitor program button in the program's menu bar.

5.5 Building and Downloading

With the configuration and logic programming complete, use EZ Design Studio to:

- 1 Build a complete control application consisting of tasks linked to compiled application programs, then

- 2 Download the software to the Control Node(s) for execution.

5.6 Running and Monitoring

Next, use EZ Design Studio or EZ Monitor to start operation. The EZ Node Status Monitor and EZ Manager can also be used to monitor control node activity, including: node and task operation (run/halted); fault conditions; node loading; and task properties (execution time, cycle time, etc.).

5.7 Using the Watch Window

A Watch Window can be launched to examine I/O, runtime symbol, global symbol, and local symbol values. The Watch Window can also set, force, and unforce symbol values. Using the application program editors it is possible to monitor program execution. After you have validated the operation and performance of your application programs, your project is complete.

6 EZ Design Studio Tutorial

In This Chapter

➤ EZ Studio	47
➤ Configuring the Node	49
➤ Creating a Simple Program	49
➤ Creating a Task.....	52
➤ Building, Downloading, and Running the Program.....	54
➤ Monitoring the Program.....	54
➤ Conclusion	57

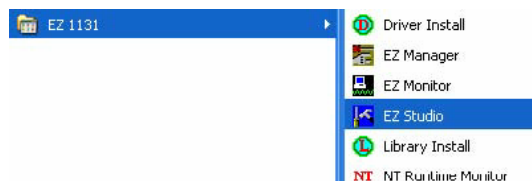
The following tutorial is presented to assist you in learning to use the EZ Design Studio development system. In this tutorial you will learn about the following:

- Starting and learning about EZ Design Studio
- Creating a control node
- Creating a simple program and task
- Building, download, and running the program
- Monitoring the program

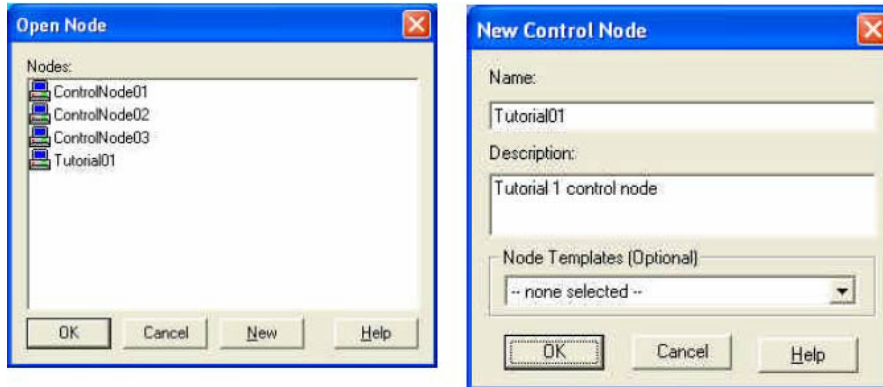
6.1 EZ Studio

EZ Design Studio is one of the major software components of EZ Design Studio. With EZ Design Studio, you design programs, build and download them to the control node, and monitor their operation. More than one EZ Design Studio instance can be open at the same time, but each EZ Studio displays the contents of a single control node. The other major software component is EZ Manager. EZ Manager is used to perform project level operations and provides a view of all control nodes, libraries, and device drivers in the project.

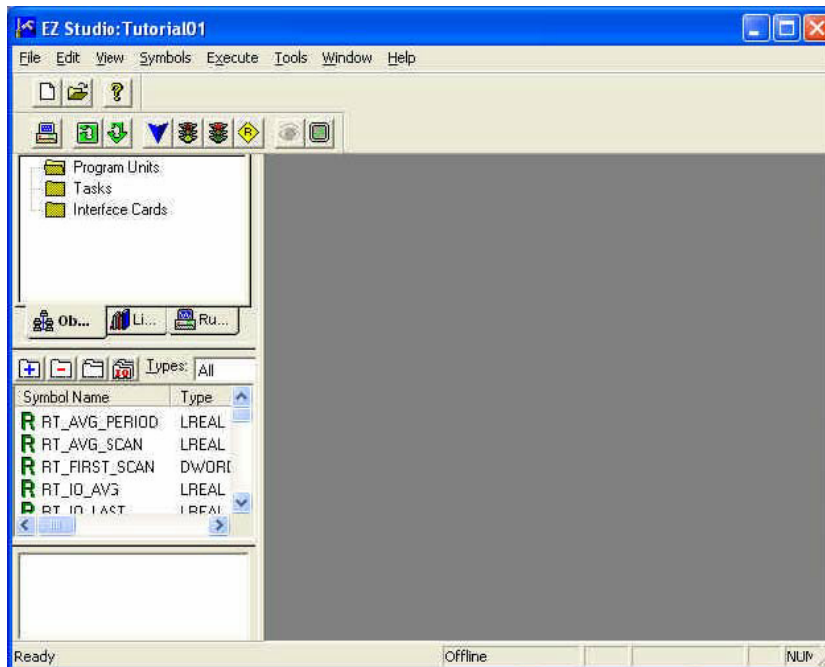
- 1 Start EZ Design Studio from the Windows Start menu. If you used the default installation path, go to the Programs menu, then EZ Design Studio and click on EZ Studio.



- When you start EZ Design Studio in this way, you are prompted to open an existing control node, or you can create a new one. For this tutorial, you will create a new control node. Click on **New**, then type in the name Tutorial01 for the name and Tutorial 1 control node for the description. For now, ignore the Node Templates field and leave it blank. Click **OK** when done.



- The control node Tutorial01 should now appear in the list of nodes that can be opened. Double-click on **Tutorial01** to open it – the EZ Studio screen should appear as in the following figure.



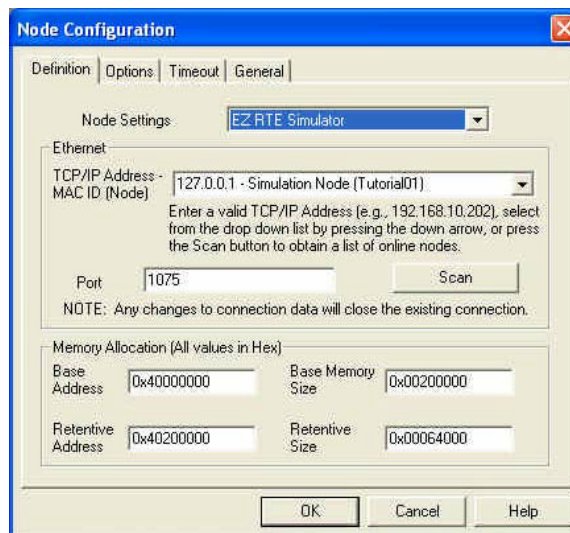
- Because this is a new control node, EZ Studio opens in a default environment.
- Notice that the Objects tab in the Node Workspace has folders to store Program Units, Tasks, and Interface Cards. These are empty right now but will display their contents as you add control objects. Also notice that the Symbol Editor shows several symbols all prefixed with "RT". These are

system symbols that are automatically included with each control node (For more information on symbols refer to the EZ Design Studio online help). The lower window on the left is the output window and displays system messages. The blank area on the right is where the program editors will appear by default.

6.2 Configuring the Node

The first thing you should do is configure your development node to match your target control node (where the control application will actually be running).

- 1 Select File | Configure Node from the menus. The Node Configuration dialog box appears.



- 2 The default should already be set to EZ RTE Simulator, which is what you will be using. If not, select it from the Node Settings list. Leave the remaining settings as is. Click **OK** to continue. (For more information on node configuration refer to the EZ Design Studio online help.) The settings determine how the development system communicates with the control node and how the application program is loaded into its memory.

6.3 Creating a Simple Program

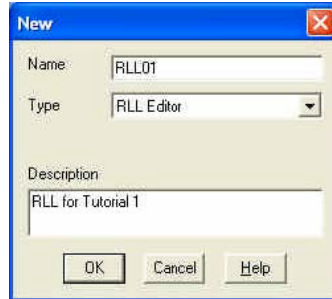
This program does not use any I/O and can be run on the local EZ-RTE Simulator. It does not do much but will quickly take you through the process of getting a node running and monitoring it.

This programming example uses an RLL (relay ladder logic) program. Do not worry if you are not familiar with RLL (or ladder diagram as it is also called).

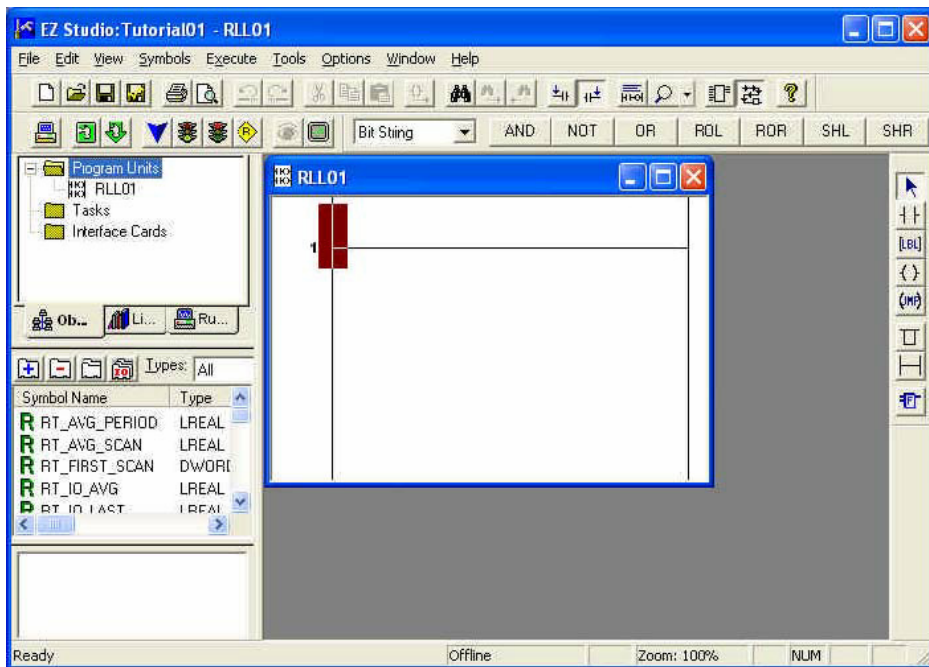
Imagine that you wish to control a motorized device that moves when you push a pushbutton. You will create a simple program to control this device.

- 1 Make sure you are on the "Objects" tab in the Node Workspace.
- 2 From the File menu, select New | Program Unit.


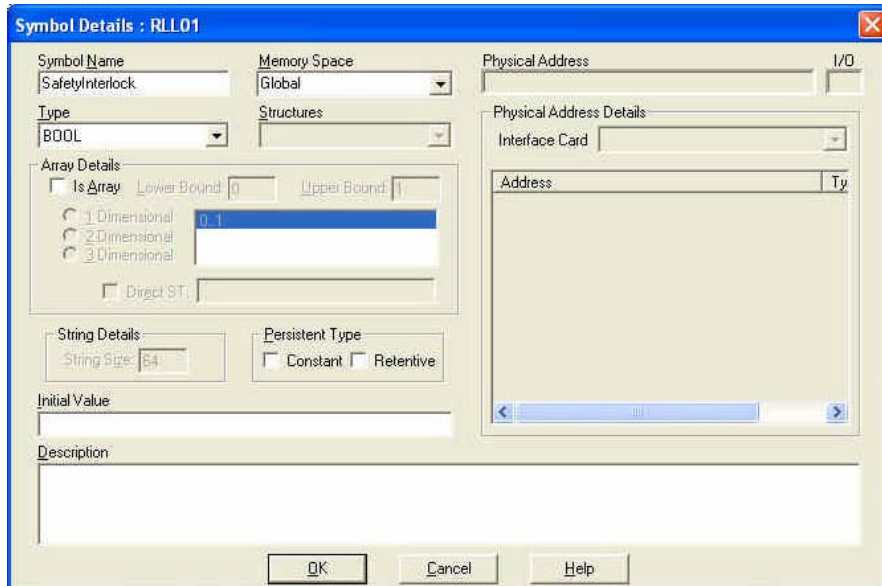
- 3 In the New dialog box, select RLL Editor from the Type list. Enter RLL01 in the Name field and RLL for Tutorial 1 in the Description field. Click **OK** to continue.



- 4 An RLL Editor window should appear in the main window area. Notice the left and right power rails and the single horizontal rung. You can resize EZ Studio and the RLL Editor window if needed. You can also see that the RLL diagram name appears under the Program Units folder.

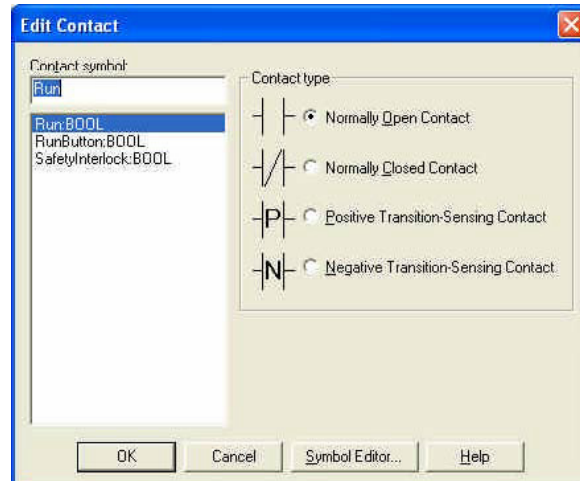


- 5 You will need a few program variables. In EZ Design Studio program variables, constants, and I/O points are collectively referred to as symbols (because you assign and reference them using a symbolic name rather than a memory location). Click on the button in the symbol editor to add a symbol. It defaults to a data type of Boolean (that is, a bit that is either on or off, true or false, high or low, etc.), which is what you want to use for all symbols in this program. Enter SafetyInterlock for the first symbol name and click **OK** to continue.

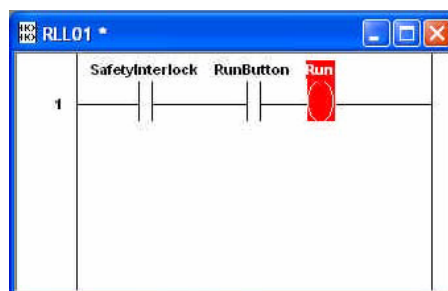
6  – Add a Symbol

- 7 Repeat and add two more symbols:
 - RunButton
 - Run
- 8 If you scroll down the list of symbols, in the symbol editor, you will now be able to locate all these symbols.
- 9 Contacts and coils are the fundamental elements of ladder logic. Contacts typically represent input points and coils represent output points. Each has an associated Boolean symbol that is mapped to a physical input or output point. (In this example, you are not using physical I/O, so Boolean variables are used for the contact and coil symbols.) The contacts and coils are placed on rungs and when the program is running, power is applied to the rung. If a (normally open) contact has its Boolean symbol TRUE, then it passes power flow. If a coil receives power flow, it sets its Boolean symbol TRUE.
- 10 Add a contact by selecting Edit | New Element | Contact from the menus. The Edit Contact dialog box appears. It lets you pick a symbol for this contact and define the contact type. Use the default of Normally Open Contact and select SafetyInterlock as the symbol (which you previously defined). Click **OK** to continue.

Note: You can also use the RLL tool bar to the right of the screen to add diagram elements.



- 11 Add another contact in the same way, this time picking RunButton as the symbol.
- 12 Add a coil by selecting Edit | New Element | Output Coil from the menus. The Edit Coil dialog box appears. It lets you pick a symbol for this coil and define the coil type. Use the default of Output Coil and select Run as the symbol (which you previously defined). Click **OK** to continue.
- 13 When done, your screen should look something like the following figure. If your contacts or coil are not in the correct order, you can select them with the mouse and drag-and-drop them to the correct spot.
- 14 Save and validate the program by selecting File | Save and Validate. If successful, you should see a File compiled successfully in message in the output window.

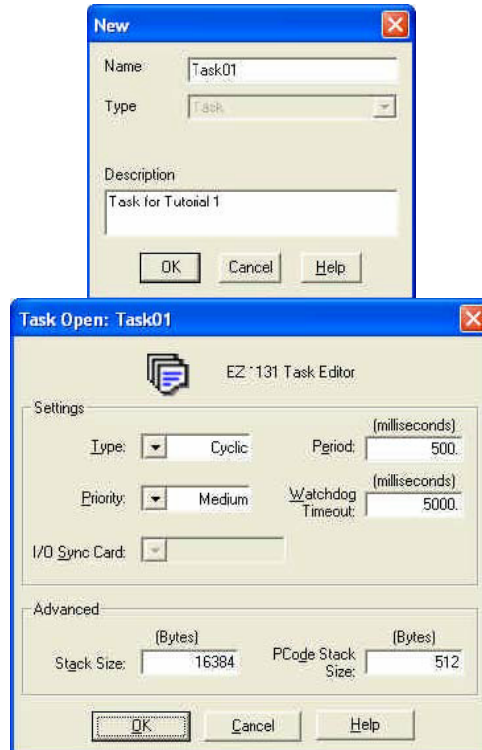


6.4 Creating a Task

The way the control system works (by IEC 61131-3 definition) is that **tasks** are the objects that are actually executed. You can think of a task as a container that holds program units (actually, an instance of the program unit). A program unit must be instantiated within a task for the instructions in that program unit to be

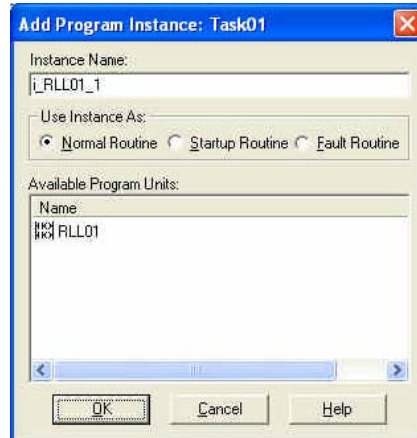
executed. Each task can have many program instantiations. Using tasks adds flexibility to control systems in that you can change task parameters, such as how often the task is run, without having to change your programs. It also lets you use the same program unit more than once.

- 1 Create a new task by selecting File | New | Task from the menus. The New dialog box appears. Enter Task01 for the task name and Task for Tutorial 1 for the description. Click **OK** to continue.
- 2 The Task Editor appears. Leave the default settings and click **OK** to continue.



- 3 Now add your RLL diagram as a task instance. Select Task01 in the Tasks folder, then select File | New | Task Instance from the menus. The Add Program Instance dialog box appears. Leave the instance as a Normal Routine, select RLL01 as the instance to add (it is the only one there), and click **OK** to continue.

Note: The Instance Name is prefixed with a "i_" to denote that this control object is a task instance and that it is suffixed with "_1" to denote which instance it is. You can actually have more than one instance of a program unit in a task. You will also see the task instance name is now positioned under Task01 in the Task folder.



6.5 Building, Downloading, and Running the Program

- 1 Because you will be using the EZ-RTE Simulator on your PC, you must start it before you download to it. Choose Tools | Start EZ-RTE Simulator from the menus. You will see a traffic light icon appear in your system tray.
- 2 Building puts all the parts of your application program together. To build your application, select Execute | Build from the menus. You will see a number of messages appear in the output window. If all goes well, the last message will be Build completed successfully.
- 3 Now you are ready to download. Select Execute | Download from the menus. You should see a Node download successful message in the output window.
- 4 To run your program, select Execute | Start from the menus. The traffic light icon in the system tray should appear with a green light now. Other than that, you do not have an indication of what your program is doing yet.

6.6 Monitoring the Program

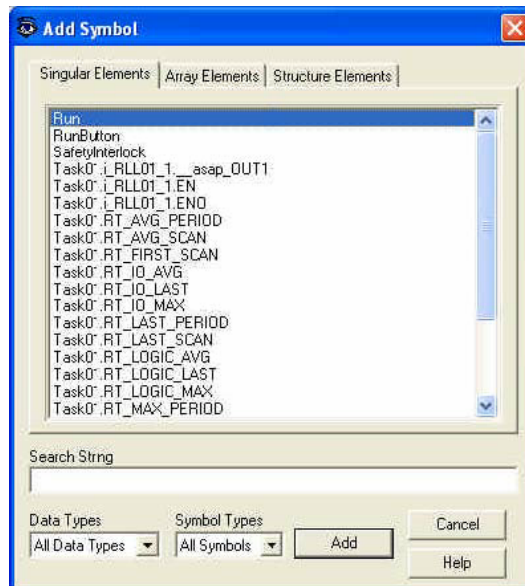
Monitoring lets you see what your application program is doing with respect to the symbol values. In RLL you can see the states of the contacts and coils: a green highlight on a contact indicates its symbol is in a state that enables it and it is passing power. Green highlight on a coil indicates it is energized. (These are the default colors.)

- 1 To monitor the RLL application program, select Execute | Online Monitoring from the menus. With monitoring on, you still do not see much in this application because the symbol values for the contacts are initialized to FALSE.

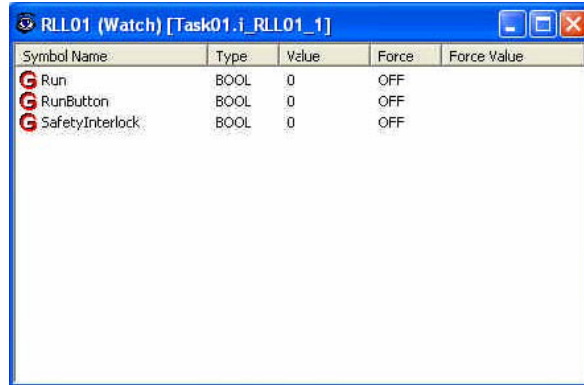
- 2 Select Execute | Watch Window from the menus. A Watch Window dialog box should appear in the screen. Reposition it if necessary (or select Tile or Cascade from the Window menu if it is not visible).



- 3 Add symbols to the Watch Window by selecting Watch | Add from the menus. The Watch Window Add Symbol dialog box appears. (Alternately you can drag and drop symbols from the Symbol Editor to add them to the Watch Window.)



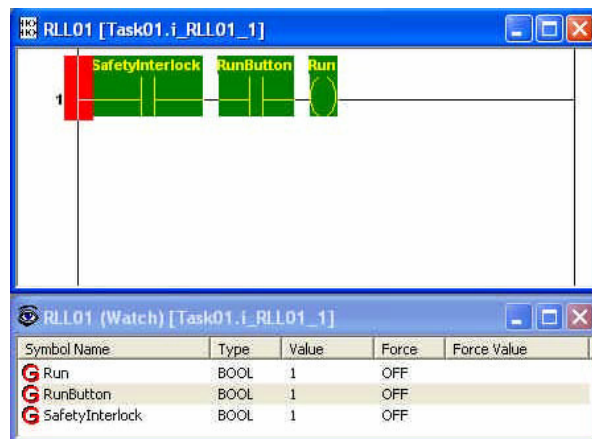
- 4 Use the mouse to select the Run, RunButton, and SafetyInterlock symbols. Click **Add** to continue. Ignore the other symbols that appear, some of these are the system runtime symbols and others are task symbols that are automatically created. The Watch Window should now look like the following figure.



- 5 Select the SafetyInterlock symbol in the Watch Window, then select Watch | Set Value from the menus. The Set Value dialog box appears. Enter 1 for the value and click **OK**. Follow the same procedure for the RunButton symbol to set its value to 1.



- 6 You should now see the SafetyInterlock contact in the RLL diagram highlight in green. After doing this, the Run output coil should automatically highlight green and its symbol value in the Watch Window change to 1, because it now has power applied.



6.7 Conclusion

That's your first EZ Design Studio application program. There are many more features to describe. You will also find easier and alternative ways of doing things using the toolbars and context menus.

When you are done with the tutorial program, you can shut down the EZ-RTE Simulator by right-clicking on the traffic light icon in the system tray and selecting Exit.

Support, Service & Warranty

ProSoft Technology, Inc. survives on its ability to provide meaningful support to its customers. Should any questions or problems arise, please feel free to contact us at:

Internet	Web Site: http://www.prosoft-technology.com/support E-mail address: support@prosoft-technology.com
Phone	+1 (661) 716-5100 +1 (661) 716-5101 (Fax)
Postal Mail	ProSoft Technology, Inc. 1675 Chester Avenue, Fourth Floor Bakersfield, CA 93301

Before calling for support, please prepare yourself for the call. In order to provide the best and quickest support possible, we will most likely ask for the following information:

- 1 Product Version Number
- 2 System architecture
- 3 Module configuration and contents of configuration file
- 4 Module Operation
 - Configuration/Debug status information
 - LED patterns
- 5 Information about the processor and user data files as viewed through the processor configuration software and LED patterns on the processor
- 6 Details about the serial devices interfaced

An after-hours answering system allows pager access to one of our qualified technical and/or application support engineers at any time to answer the questions that are important to you.

Module Service and Repair

The 7000-ADM device is an electronic product, designed and manufactured to function under somewhat adverse conditions. As with any product, through age, misapplication, or any one of many possible problems the device may require repair.

When purchased from ProSoft Technology, Inc., the device has a 1 year parts and labor warranty (3 years for RadioLinx) according to the limits specified in the warranty. Replacement and/or returns should be directed to the distributor from whom the product was purchased. If you must return the device for repair, obtain an RMA (Returned Material Authorization) number from ProSoft Technology, Inc. Please call the factory for this number, and print the number prominently on the outside of the shipping carton used to return the device.

General Warranty Policy – Terms and Conditions

ProSoft Technology, Inc. (hereinafter referred to as ProSoft) warrants that the Product shall conform to and perform in accordance with published technical specifications and the accompanying written materials, and shall be free of defects in materials and workmanship, for the period of time herein indicated, such warranty period commencing upon receipt of the Product. Limited warranty service may be obtained by delivering the Product to ProSoft in accordance with our product return procedures and providing proof of purchase and receipt date. Customer agrees to insure the Product or assume the risk of loss or damage in transit, to prepay shipping charges to ProSoft, and to use the original shipping container or equivalent. Contact ProSoft Customer Service for more information.

This warranty is limited to the repair and/or replacement, at ProSoft's election, of defective or non-conforming Product, and ProSoft shall not be responsible for the failure of the Product to perform specified functions, or any other non-conformance caused by or attributable to: (a) any misuse, misapplication, accidental damage, abnormal or unusually heavy use, neglect, abuse, alteration (b) failure of Customer to adhere to ProSoft's specifications or instructions, (c) any associated or complementary equipment, software, or user-created programming including, but not limited to, programs developed with any IEC1131-3 programming languages, 'C' for example, and not furnished by ProSoft, (d) improper installation, unauthorized repair or modification (e) improper testing, or causes external to the product such as, but not limited to, excessive heat or humidity, power failure, power surges or natural disaster, compatibility with other hardware and software products introduced after the time of purchase, or products or accessories not manufactured by ProSoft; all of which components, software and products are provided as-is. In no event will ProSoft be held liable for any direct or indirect, incidental consequential damage, loss of data, or other malady arising from the purchase or use of ProSoft products.

ProSoft's software or electronic products are designed and manufactured to function under adverse environmental conditions as described in the hardware specifications for this product. As with any product, however, through age, misapplication, or any one of many possible problems, the device may require repair.

ProSoft warrants its products to be free from defects in material and workmanship and shall conform to and perform in accordance with published technical specifications and the accompanying written materials for up to one year (12 months) from the date of original purchase (3 years for RadioLinx products) from ProSoft. If you need to return the device for repair, obtain an RMA (Returned Material Authorization) number from ProSoft Technology, Inc. in accordance with the RMA instructions below. Please call the factory for this number, and print the number prominently on the outside of the shipping carton used to return the device.

If the product is received within the warranty period ProSoft will repair or replace the defective product at our option and cost.

Warranty Procedure: Upon return of the hardware product ProSoft will, at its option, repair or replace the product at no additional charge, freight prepaid, except as set forth below. Repair parts and replacement product will be furnished on an exchange basis and will be either reconditioned or new. All replaced product and parts become the property of ProSoft. If ProSoft determines that the Product is not under warranty, it will, at the Customer's option, repair the Product using then current ProSoft standard rates for parts and labor, and return the product freight collect.

Limitation of Liability

EXCEPT AS EXPRESSLY PROVIDED HEREIN, PROSOFT MAKES NO WARRANTY OF ANY KIND, EXPRESSED OR IMPLIED, WITH RESPECT TO ANY EQUIPMENT, PARTS OR SERVICES PROVIDED PURSUANT TO THIS AGREEMENT, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. NEITHER PROSOFT OR ITS DEALER SHALL BE LIABLE FOR ANY OTHER DAMAGES, INCLUDING BUT NOT LIMITED TO DIRECT, INDIRECT, INCIDENTAL, SPECIAL OR CONSEQUENTIAL DAMAGES, WHETHER IN AN ACTION IN CONTRACT OR TORT (INCLUDING NEGLIGENCE AND STRICT LIABILITY), SUCH AS, BUT NOT LIMITED TO, LOSS OF ANTICIPATED PROFITS OR BENEFITS RESULTING FROM, OR ARISING OUT OF, OR IN CONNECTION WITH THE USE OR FURNISHING OF EQUIPMENT, PARTS OR SERVICES HEREUNDER OR THE PERFORMANCE, USE OR INABILITY TO USE THE SAME, EVEN IF ProSoft OR ITS DEALER'S TOTAL LIABILITY EXCEED THE PRICE PAID FOR THE PRODUCT.

Where directed by State Law, some of the above exclusions or limitations may not be applicable in some states. This warranty provides specific legal rights; other rights that vary from state to state may also exist. This warranty shall not be applicable to the extent that any provisions of this warranty are prohibited by any Federal, State or Municipal Law that cannot be preempted. Contact ProSoft Customer Service at +1 (661) 716-5100 for more information.

RMA Procedures

In the event that repairs are required for any reason, contact ProSoft Technical Support at +1 661.716.5100. A Technical Support Engineer will ask you to perform several tests in an attempt to diagnose the problem. Simply calling and asking for a RMA without following our diagnostic instructions or suggestions will lead to the return request being denied. If, after these tests are completed, the module is found to be defective, we will provide the necessary RMA number with instructions on returning the module for repair.

Index

A

Application Development Overview • 39
AppSrv CE Features • 7
AppSrv Configuration Utility (TCU) • 13
AppSrv_Full WinCE 4.2 Platform Release
Notes • 10

B

Booting WinCE • 11
Building and Downloading • 45
Building, Downloading, and Running the
Program • 54

C

CE Components • 11
Conclusion • 57
Configuring and Programming • 39
Configuring the Node • 49
Controller • 37
Controller Spacing • 20
Creating a Simple Program • 49
Creating a Task • 52

D

Development Station • 37
DIN-rail • 27
Display Properties • 11

E

Embedded (Physical Control Node) • 40
Environmental Specifications • 22
EZ Design Studio • 33
EZ Design Studio Applications • 38
EZ Design Studio Features • 35
EZ Design Studio Software Components • 34
EZ Design Studio Tutorial • 47
EZ Manager • 40
EZ Studio • 40, 47

F

Flowchart • 34
For Rockwell Automation Compact I/O • 32
For Rockwell Automation Flex I/O • 32

G

Grounding the Controller • 21

I

Important Installation Instructions • 3
Important User Information • 18
Inserting/Removing a CAM • 24
Installing AppSrvCE • 23
Installing EZ Design Studio • 37
Installing the CAM • 30
Introducing the System • 7
Introduction • 7, 10

K

Known Issues • 14

L

Launching an Application at Startup - Batch
File Method • 14
Launching an Application at Startup -
Registry Method • 14
Logic Control Programs • 45

M

Miswiring Power Connector • 20
Monitoring the Program • 54
Mounting the Unit • 25

P

Panel Mount • 26
Planning • 44
Platform Requirements • 37
Please Read This Notice • 2
Post-Installation • 37
Power Supply Requirements • 31
Preventing Damage from Electrostatic
Discharge • 18
Procedures • 40

R

Registry Lock • 12
Registry Persistence • 12
Removing the Base Unit from the DIN Rail •
31
Revision History • 14
Running and Monitoring • 46
Running and Operating • 39
Running EZ Design Studio • 43
Runtime Connection • 38

S

- Scheduling Program Execution • 45
- Startup and System Configuration • 44
- Storage Devices • 12
- Support, Service & Warranty • 59
- Supported Devices • 10
- Surge Protection for AppSrvCE Power Input
• 18

U

- Unpacking the Base Unit • 23
- Using the Watch Window • 46
- Utility API • 13

V

- Viewing the Base Unit • 24

W

- What is EZ Design Studio? • 33
- Where to Get More Information • 38
- Wiring Recommendation • 19
- Wiring Requirements • 19
- Wiring the Base Unit Power Supply • 32

Y

- Your Feedback Please • 2