# SCADAPack E Target 5
# Technical Reference


Documentation

# Table of Contents

# I SCADAPack E Target 5 Technical Reference

Schneider Electric

Documentation



Version: 8.05.4

The information provided in this documentation contains general descriptions and/or technical characteristics of the performance of the products contained herein. This documentation is not intended as a substitute for and is not to be used for determining suitability or reliability of these products for specific user applications. It is the duty of any such user or integrator to perform the appropriate and complete risk analysis, evaluation and testing of the products with respect to the relevant specific application or use thereof. Neither Schneider Electric nor any of its affiliates or subsidiaries shall be responsible or liable for misuse of the information contained herein. If you have any suggestions for improvements or amendments or have found errors in this publication, please notify us.

No part of this document may be reproduced in any form or by any means, electronic or mechanical, including photocopying, without express written permission of Schneider Electric.

All pertinent state, regional, and local safety regulations must be observed when installing and using this product. For reasons of safety and to help ensure compliance with documented system data, only the manufacturer should perform repairs to components.

When devices are used for applications with technical safety requirements, the relevant instructions must be followed. Failure to use Schneider Electric software or approved software with our hardware products may result in injury, harm, or improper operating results.

Failure to observe this information can result in injury or equipment damage.

## 1 Technical Support

Support related to any part of this documentation can be directed to one of the following support centers.

### Technical Support: The Americas

Available Monday to Friday 8:00am – 6:30pm Eastern Time

Toll free within North America 1-888-226-6876

| | |
|---|---|
| Direct Worldwide | +1-613-591-1943 |
| Email | TechnicalSupport@controlmicrosystems.com |

### Technical Support: Europe

Available Monday to Friday 8:30am – 5:30pm Central European Time

| | |
|---|---|
| Direct Worldwide | +31 (71) 597-1655 |
| Email | euro-support@controlmicrosystems.com |

### Technical Support: Asia

Available Monday to Friday 8:00am – 6:30pm Eastern Time (North America)

| | |
|---|---|
| Direct Worldwide | +1-613-591-1943 |
| Email | TechnicalSupport@controlmicrosystems.com |

### Technical Support: Australia

| | |
|---|---|
| Inside Australia | 1300 369 233 |
| Email | au.help@schneider-electric.com |

## 2 Safety Information

Read these instructions carefully, and look at the equipment to become familiar with the device before trying to install, operate, or maintain it. The following special messages may appear throughout this documentation or on the equipment to warn of potential hazards or to call attention to information that clarifies or simplifies a procedure.

| | |
|---|---|
|  | The addition of this symbol to a Danger or Warning safety label indicates that an electrical hazard exists, which will result in personal injury if the instructions are not followed. |

| | |
|---|---|
|  | This is the safety alert symbol. It is used to alert you to potential personal injury hazards. Obey all safety messages that follow this symbol to avoid possible injury or death. |

| **⚠ DANGER** |
|---|
| **DANGER** indicates an imminently hazardous situation which, if not avoided, **will result in** death or serious injury. |

## ⚠WARNING

**WARNING** indicates a potentially hazardous situation which, if not avoided, **can result** in death or serious injury.

## ⚠CAUTION

**CAUTION** indicates a potentially hazardous situation which, if not avoided, **can result** in minor or moderate injury.

## CAUTION

**CAUTION** used without the safety alert symbol, indicates a potentially hazardous situation which, if not avoided, **can result in** equipment damage..

### PLEASE NOTE

Electrical equipment should be installed, operated, serviced, and maintained only by qualified personnel. No responsibility is assumed by Schneider Electric for any consequences arising out of the use of this material.

A qualified person is one who has skills and knowledge related to the construction and operation of electrical equipment and the installation, and has received safety training to recognize and avoid the hazards involved.

### BEFORE YOU BEGIN

SCADAPack Workbench and SCADAPack E Smart RTU are not suitable for controlling safety-critical systems. SCADAPack Workbench and SCADAPack E Smart RTU are not tested for, nor have approval for use in, the control of safety-critical systems. Safety-critical systems should be controlled by an approved safety-critical platform that is independent of SCADAPack Workbench and SCADAPack E Smart RTU.

## ⚠WARNING

**UNINTENDED EQUIPMENT OPERATION**

Do not control safety-critical systems with SCADAPack Workbench and SCADAPack E Smart RTU.

**Failure to follow these instructions can result in death, serious injury or equipment damage.**

Do not use this product on machinery lacking effective point-of-operation guarding. Lack of effective point-of-operation guarding on a machine can result in serious injury to the operator

of that machine.

| ⚠CAUTION |
| --- |
| **EQUIPMENT OPERATION HAZARD** |
| • Verify that all installation and set up procedures have been completed.<br>• Before operational tests are performed, remove all blocks or other temporary holding means used for shipment from all component devices.<br>• Remove tools, meters, and debris from equipment. |
| **Failure to follow these instructions can result in injury or equipment damage.** |

Follow all start-up tests recommended in the equipment documentation. Store all equipment documentation for future references.

Software testing must be done in both simulated and real environments.

Verify that the completed system is free from all short circuits and grounds, except those grounds installed according to local regulations (according to the National Electrical Code in the U.S.A, for instance). If high-potential voltage testing is necessary, follow recommendations in equipment documentation to prevent accidental equipment damage.

Before energizing equipment:

- Remove tools, meters, and debris from equipment.
- Close the equipment enclosure door.
- Remove ground from incoming power lines.
- Perform all start-up tests recommended by the manufacturer.

## OPERATION AND ADJUSTMENTS

The following precautions are from the NEMA Standards Publication ICS 7.1-1995 (English version prevails):

- Regardless of the care exercised in the design and manufacture of equipment or in the selection and ratings of components, there are hazards that can be encountered if such equipment is improperly operated.
- It is sometimes possible to misadjust the equipment and thus produce unsatisfactory or unsafe operation. Always use the manufacturer's instructions as a guide for functional adjustments. Personnel who have access to these adjustments should be familiar with the equipment manufacturer's instructions and the machinery used with the electrical equipment.
- Only those operational adjustments actually required by the operator should be accessible to the operator. Access to other controls should be restricted to prevent unauthorized

changes in operating characteristics.

# 3 Preface

## Purpose

The purpose of this document is to describe in detail the interface between the SCADAPack E Smart RTU and the SCADAPack Workbench software.

It is assumed that the reader is familiar with the SCADAPack Workbench software. For more information about the Workbench program environment, see the Workbench Help documentation.

Additional programming options specific to the SCADAPack E Smart RTU can be found in the *SCADAPack E Target 5 Function Blocks Reference* manual.

The Workbench Help documentation is not dedicated to any particular target and describes general concepts. The material covered in this manual together with the *SCADAPack E Target 5 Function Blocks Reference* manual therefore supplements the Workbench Help manuals for it deals exclusively with the implementation of the IEC 61131-3 Target software on a Schneider Electric SCADAPack E Smart RTU.

## Assumed Knowledge

Familiarity with the SCADAPack Workbench is strongly recommended.

## Target Audience

- Systems Engineers
- Commissioning Engineers
- Maintenance Technicians

## References

- Workbench Help
- SCADAPack E Target 5 Help

# 4 Overview & Terminology

## Overview

This document describes the interface between SCADAPack E Smart RTU operating and SCADAPack Workbench software.

ISaGRAF provides IEC61131-3 international standard PLC programming capability to the SCADAPack E Smart RTU.

The IEC 61131-3 target Virtual Machines (VM's) execute on the SCADAPack E Smart RTU and are built in to the operating system firmware.

The SCADAPack Workbench software executes on a PC and provides application generation,

transferring and debugging through connection to the target by serial or network communications.

The SCADAPack E Smart RTU provides two IEC 61131-3 target types, *Target 3* and *Target 5*. This document covers Target 5 and SCADAPack Workbench.

## Highlights

By way of summary, the following facilities are notable Workbench facilities, or are provided in addition to the standard Workbench facilities when used on SCADAPack E Smart RTU:

- Load applications to RTU via SCADAPack Workbench, via File Transfer, remotely via DNP3, remotely via IEC60870-5-101 & -104 protocol
- Remote debugging of IEC 61131-3 applications via serial or TCP/IP connections
- Dual IEC 61131-3 target Virtual Machines (VM's) for executing 2 separate application resources simultaneously
- Download IEC 61131-3 application project source to the RTU, for later upload
- Permits Online Changes to running application resources
- Scan control prioritization for improved performance or high accuracy scan intervals
- Interaction between IEC 61131-3 applications and RTU operating system facilities
- Interaction between IEC 61131-3 applications and RTU point database attributes
- IEC 61131-3 application interaction with RTU database floating point numeric and string data types
- IEC 61131-3 application application control over data exchange with peer RTU nodes
- IEC 61131-3 application control over trend sampling
- IEC 61131-3 application file management
- IEC 61131-3 application custom communication protocol management
- IEC 61131-3 application interface to common PLC and peripheral device protocols

## Terminology

The SCADAPack E Smart RTU provides two IEC 61131-3 target types, called Target 3 and Target 5. Target 3 is used with 16-bit ISaGRAF Workbench (not covered by this document) and Target 5 with SCADAPack Workbench (covered by this document).

A IEC 61131-3 application is called a Solution in Workbench.

A Solution for a IEC 61131-3 application contains a Project.

A Project contains a Device, which a SCADAPack E Smart RTU. The example above shows a SCADAPack 300E RTU.

A Device can contain one or two Resources. Resources are independent parts of the application. Resources can be stopped, downloaded, uploaded, and started independently. A Resource contains IEC 61131-3 programs (also called Program Organization Units or POU), I/O devices, functions, function blocks, and variables. Resources can exchange information through variable binding. These are explained in the sections that follow.

This solution structure is used for SCADAPack Workbench applications. Although it's possible to structure the solution in other ways, they are not supported by the SCADAPack E Smart RTU.

The two Resources are independent, but can exchange data through variable *binding*.

## 4.1 SCADAPack Workbench Software

An IEC 61131-3 application resource can be loaded, configured & debugged on the SCADAPack E RTU from a standard PC running the SCADAPack Workbench if there is a connection between the PC and the RTU. Serial and Ethernet / TCP/IP communication facilities are provided by the RTU to allow different options of connection from a standard PC.

The SCADAPack Workbench software is used to create, manage and simulate IEC 61131-3 applications.

The Workbench can transfer IEC 61131-3 resources to the target and provides application-debugging facilities.

Workbench communication with SCADAPack E RTUs is supported for Windows XP with SP3 (with Power User privileges), Windows Server 2003, Windows Vista, Windows Server 2008, or Windows 7 operating system. For more information on the SCADAPack Workbench, refer to the Workbench Help documentation.

SCADAPack Workbench can connect to an SCADAPack E RTU in a number of ways:

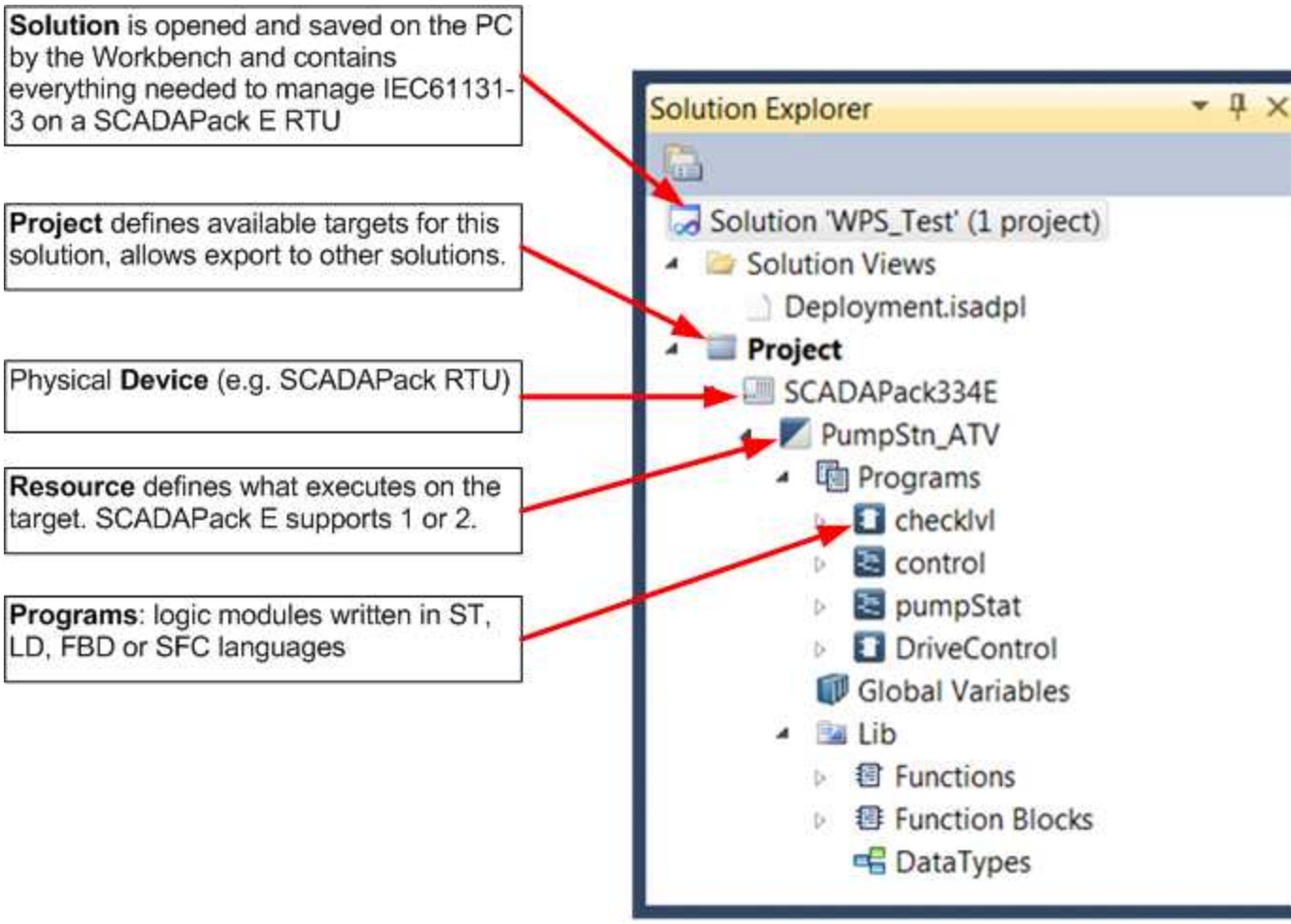
- Local serial port communications. This is called *IsaRSI* after the name of the network driver that provides communication with an IEC 61131-3 target on a serial port.
- Ethernet communications via a TCP/IP port. The Enhanced TCP/IP protocol, *ETCP* is the network driver used for communication with an IEC 61131-3 target on Ethernet.

The type of connection used by the Workbench is a function of the project configuration. The SCADAPack Workbench software offers pre-configured ***Project Templates*** 12 for either Serial or TCP/IP connections.

Only one active connection is used to communicate with either running VM.

The ***Target Remote Access*** 80 section describes the various connection options that are available between the SCADAPack Workbench and SCADAPack E RTU's.

The hierarchy of a SCADAPack Workbench IEC61131-3 *Solution* is shown in the diagram below.

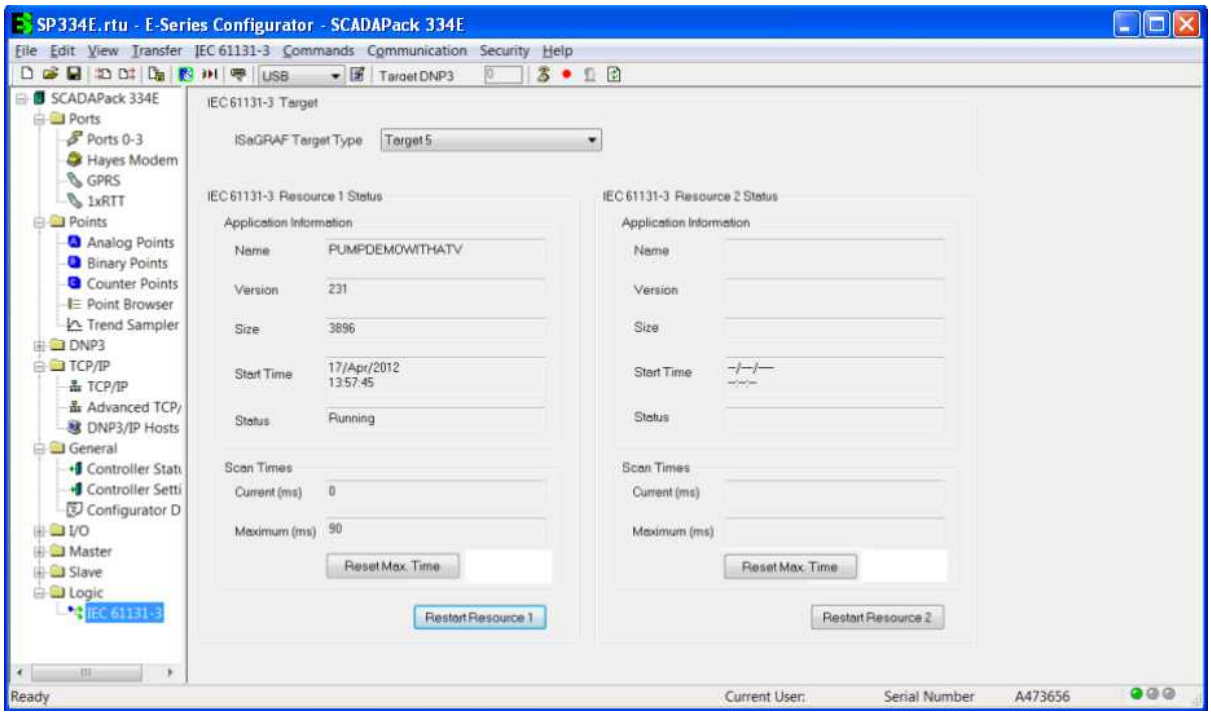



## 4.2 Target & Target Memory Usage

### SCADAPack E IEC 61131-3 Target

The SCADAPack E RTU is equipped with ISaGRAF IEC 61131-3 target software. This allows the RTU to perform PLC control functions using the IEC 61131-3 international standard. The control functions provided by ISaGRAF targets are completely autonomous of any supervisory (SCADA Master) system or communications network (e.g. DNP3). The IEC 61131-3 application operates on the RTU regardless of the state of remote communications.

The SCADAPack E RTU provides two IEC 61131-3 target types. Target 3 allows compatibility with IEC61131-3 applications developed using the ISaGRAF 3 Workbench (16-bit). Target 5 allows applications to be developed using the SCADAPack Workbench. Only one IEC 61131-3 target type can be active at a time in the SCADAPack E RTU. The *ISaGRAF Target Type* setting activates the appropriate IEC 61131-3 target type. This document describes the use of Target 5.

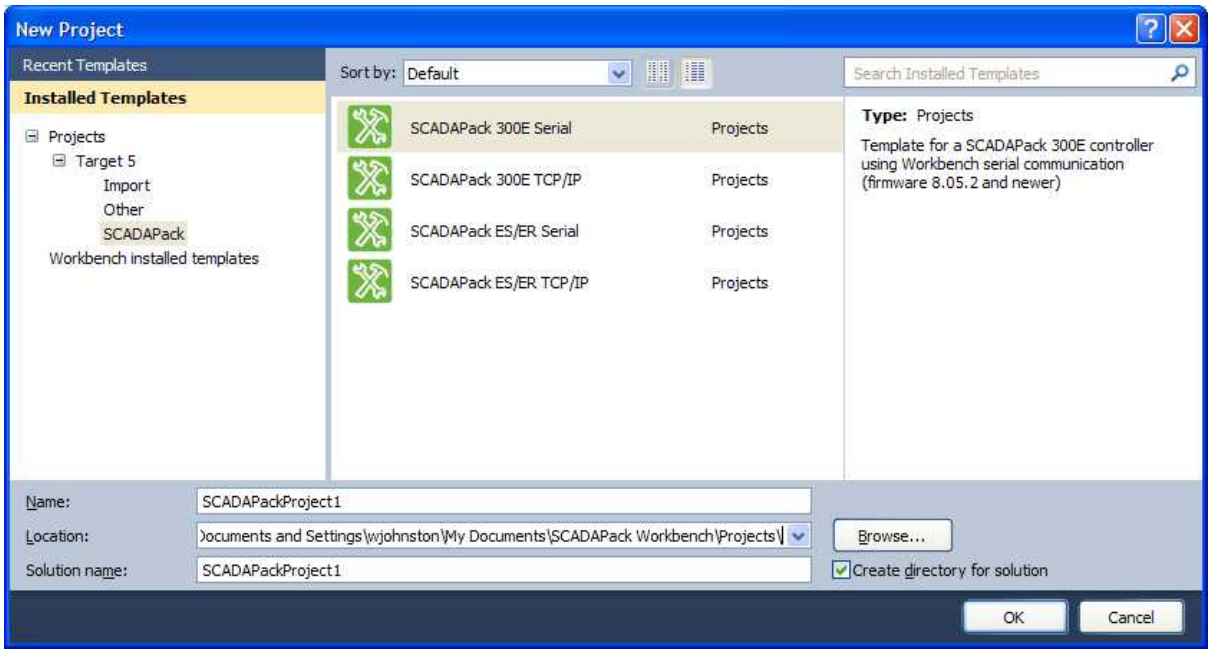



*SCADAPack E Configurator IEC 61131-3 Page*

The SCADAPack E RTU firmware supports the simultaneous execution of up to **two IEC 61131-3 target VM's** (Virtual Machines) on the same RTU. This allows up to two independent applications resources to execute simultaneously on the same RTU. The two IEC 61131-3 targets within the RTU have a fixed Workbench *Resource Number* of 1 and 2 respectively. Both IEC 61131-3 target VM's are activated by default, i.e. there is no additional configuration required to activate the second VM.

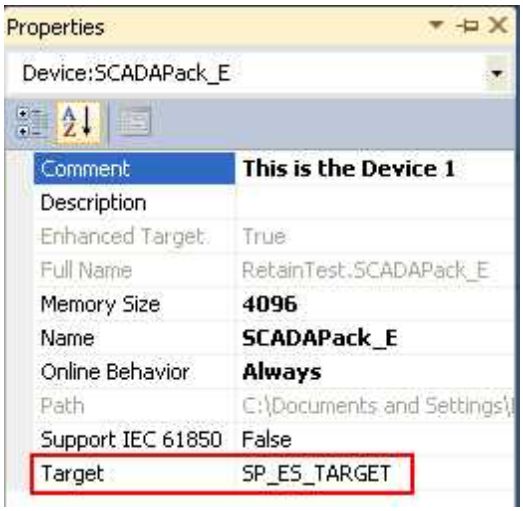
The status (running or halted) of an IEC 61131-3 resource on each VM can be obtained from the ***SCADAPack E Configurator*** 12 software or via separate *Application Halted* system binary points 50100 and 50101. These binary points are part of the system points dedicated to providing the status of the RTU. For more information see the *SCADAPack E Operational Reference Manual*.

The SCADAPack E target device type is selected from one of several project *templates* selected from the Workbench *New Project* window. There are currently two templates for SCADAPack 300E models and two for SCADAPack ES/ER models, depending on the the default method of communication between the SCADAPack Workbench and the IEC 61131-3 target on the RTU.




*New Project Template selection*

For an existing project, the current target type can be found by opening the *Properties* window on the project *Device* in the *Solution Explorer* window.




*Workbench Device Properties window*

The Workbench Resource *Embedded Symbol Table* property is not required to be enabled for the SCADAPack E RTU.

The Device Properties window allows the Target to be changed to SIMULATOR. Do not change the target setting. Rather, select Simulator from the debugging toolbar to run the solution on a simulator. The SIMULATOR target type is present only to allow the Simulator to function, and should not be selected from the Properties window.

### Target Memory Usage

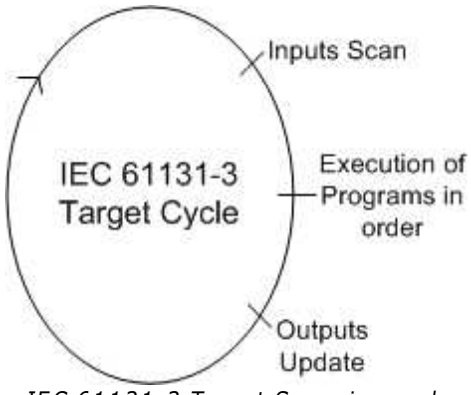
The SCADAPack E Smart RTU is equipped with FLASH memory, which contains the RTU operating system, IEC 61131-3 target kernel engine and telemetry communications software. The RTU is also equipped with battery-backed system Static RAM for storage of IEC 61131-3 application data. Up to 4K bytes of Static RAM (per-VM) is available for IEC 61131-3 retained variables. See Section ***Retained Variables*** 32 for details.

RTU system RAM is also available for facilities such as Online IEC 61131-3 application modification and data.

See Section ***Application Storage*** 47 for details on application storage.

# 5 Target Scanning Cycle

The IEC 61131-3 target executes user applications in a cyclic fashion. As such, user applications are also designed to execute in a cyclic fashion. As shown below, the ***Target Scanning Cycle*** 14 has distinct phases. These include obtaining inputs to the IEC 61131-3 application (via Input I/O Devices), execution of IEC 61131-3 programs, and updating of outputs (via Output I/O Devices). The order of IEC 61131-3 program execution is as shown in the Program list, and can be modified by changing the "Order" setting in the program's *Properties*.




*IEC 61131-3 Target Scanning cycle*

Each Workbench Resource has automatically-defined System variables that hold the values relating to cycle count, timing, kernel bindings, and resource information. You can view System variables from the *Global Variables* dictionary instance for each resource. For example, __SYSVA_TCYCURRENT is a global TIME variable which holds the value of the current cycle time. These values are normally read with the Workbench debugger running and connected to the target. However, the variables can be also be used within an application program.

Resource1-VAR | Deployment.isadpl | Start Page

| Name | Data Type | Dimension | Alias | Comment |
|---|---|---|---|---|
| __SYSVA_CYCLECNT | DINT | | | Cycle counter |
| __SYSVA_CYCLEDATE | TIME | | | Timestamp of the beginning of the cycle in milliseconds |
| __SYSVA_KVBPERR | BOOL | | | Kernel variable binding producing error (production error) |
| __SYSVA_KVBCERR | BOOL | | | Kernel variable binding consuming error (consumption error) |
| __SYSVA_RESNAME | STRING | | | Resource name (max length=255) |
| __SYSVA_SCANCNT | DINT | | | Input scan counter |
| __SYSVA_TCYCYCTIME | TIME | | | Programmed cycle time |
| __SYSVA_TCYCURRENT | TIME | | | Current cycle time |
| __SYSVA_TCYMAXIMUM | TIME | | | Maximum cycle time since last start |
| __SYSVA_TCYOVERFLOW | DINT | | | Number of cycle overflows |
| __SYSVA_RESMODE | SINT | | | Resource execution mode |
| __SYSVA_CCEXEC | BOOL | | | Execute one cycle when application is in cycle to cycle mode |

*Workbench Resource 'System Variables'*

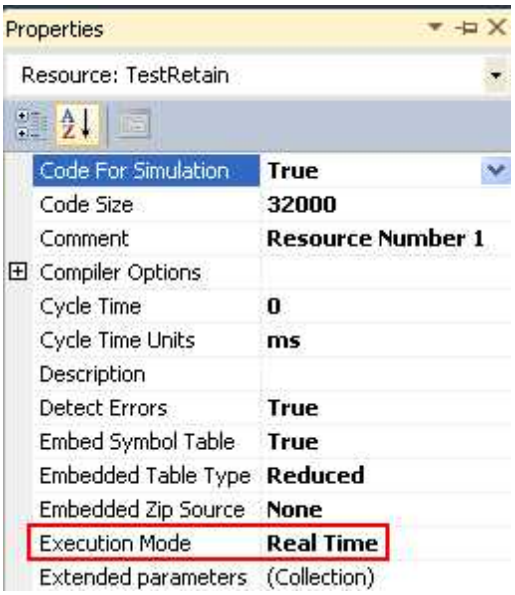
The IEC 61131-3 target executing on an SCADAPack E RTU can be configured to scan in various ways:

- ***Real Time Scanning*** 16
- ***Cycle Time Scanning*** 18
- ***High Priority Scanning*** 21

## 5.1 Continuous Scanning

This is the default *Execution Mode* that is used by IEC 61131-3 resources (application programs). This mode is enabled if the resource's *Execution Mode* property is set to *Real Time* and the *Cycle Time* is set to zero.

If enabled, at the end of an IEC 61131-3 target scan, the target simply restarts the scanning cycle. Typically, the SCADAPack E RTU has many activities it needs to perform in addition to scanning the user's IEC 61131-3 application. As such, the application may be interrupted for short periods throughout the scan cycle. This can result in some variations in the time it takes the VM to perform each scan. If the RTU becomes unusually busy performing these activities, this can result in slower execution of the user's IEC 61131-3 application.




*Resource Execution Mode property*

This mode can be temporarily overridden from within the SCADAPack Workbench debugger. The debugger can be used to change to ***Cycle Time Scanning*** 18 by writing a new value to the ***__SYSVA_TCYCYCTIME*** 18 resource global TIME variable.

**⚠WARNING**

**UNINTENDED EQUIPMENT OPERATION**

Evaluate the operational state of the equipment monitored and controlled by the RTU before debugging. Hazardous situations can occur if system state is not confirmed prior to debugging.

**Failure to follow these instructions can result in death, serious injury or equipment damage.**

If the IEC 61131-3 application or the SCADAPack E RTU is restarted, a temporary cycle timing setting is lost. The cycle time will revert to the Resource properties' *Execution Mode* and *Cycle Time* settings.

It is recommended that, where possible, IEC 61131-3 applications use continuous scanning mode and be designed to take into account variations in the timing of the scan rate.

## 5.2 Cycle Time Scanning

This scanning mode can be either permanently set for an IEC 61131-3 application resource or temporarily set. Instead of the IEC 61131-3 target restarting the scanning cycle immediately after updating the outputs, as in the default continuous mode, the scanning process is stopped for a certain time interval. This time interval, which makes up the difference between the actual time of a complete scan and a fixed cycle time, is now available for the RTU to perform activities that previously may have interrupted the scanning cycle. As such, the IEC 61131-3 application will tend to have more consistent cycle scan times.

The SCADAPack E RTU can still schedule other system operations in the middle of a IEC 61131-3 scan if necessary. This is particularly the case for larger applications. Smaller applications will not normally be interrupted by routine RTU activities.

The Workbench Debugger indicates a non-zero **__SYSVA_TCYCYCTIME** System variable value when *Cycle Time Scanning* mode is in use as shown in the figure below.

Resource1-VAR | Start Page | Deployment.isadpl

| Name | Logical Value | Physical Value | Lock | Data Type | Dimension | Alias | Comment |
|---|---|---|---|---|---|---|---|
| __SYSVA_CYCLECNT | 1230 | N/A | | DINT | | | Cycle counter |
| __SYSVA_CYCLEDATE | T#2d2h2m46s37 | N/A | | TIME | | | Timestamp of the beginning of the cycle in milliseconds |
| __SYSVA_KVBPERR | | N/A | | BOOL | | | Kernel variable binding producing error (production error) |
| __SYSVA_KVBCERR | | N/A | | BOOL | | | Kernel variable binding consuming error (consumption error) |
| __SYSVA_RESNAME | PASSWORD\5CA | N/A | | STRING | | | Resource name (max length=255) |
| __SYSVA_SCANCNT | 1228 | N/A | | DINT | | | Input scan counter |
| __SYSVA_TCYCYCTIME | T#100ms | N/A | | TIME | | | Programmed cycle time |
| __SYSVA_TCYCURRENT | T#0s | N/A | | TIME | | | Current cycle time |
| __SYSVA_TCYMAXIMUM | T#0s | N/A | | TIME | | | Maximum cycle time since last start |
| __SYSVA_TCYOVERFLOW | 0 | N/A | | DINT | | | Number of cycle overflows |
| __SYSVA_RESMODE | 3 | N/A | | SINT | | | Resource execution mode |
| __SYSVA_CCEXEC | | N/A | | BOOL | | | Execute one cycle when application is in cycle to cycle mode |

*Workbench Debugger System Variables*

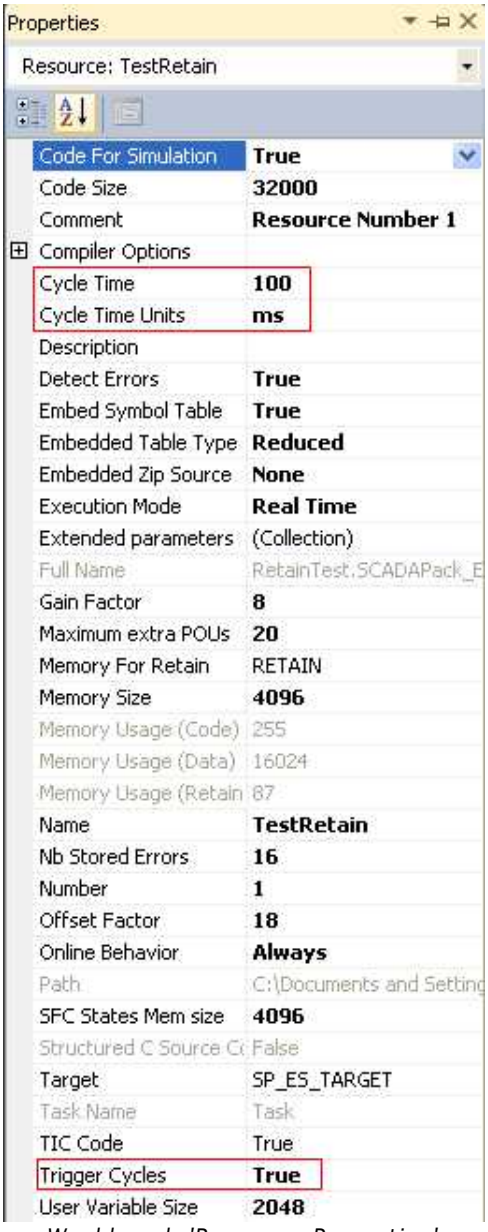
The Workbench Debugger will increment the **__SYSVA_TCYOVERFLOW** System variable value if the time taken to perform an individual scan is higher than the fixed (allowed) time. In this case the fixed *Cycle Time* should be increased.

The following sections describe methods for permanently or temporarily setting Cycle Timing scanning for an IEC 61131-3 resource. This mode can also be changed from within a user application.

### 5.2.1 Permanent & Temporary Cycle Timing

## Permanent Cycle Timing

To permanently set an IEC 61131-3 resource cycle time from the Workbench, enable *Trigger Cycles* from the Resource *Properties dialog* from the *Solution Explorer* and enter a value in the *Cycle Time* field. This needs to be done before the application *Build*.




*Workbench 'Resource Properties' Window*

To change the IEC 61131-3 scanning mode from cycle timing back to Continuous scanning, set the *Trigger Cycles* property to False, rebuild and download the application resource.

## ⚠WARNING

**UNINTENDED EQUIPMENT OPERATION**

Evaluate the operational state of the equipment monitored and controlled by the RTU before downloading. Hazardous situations can occur if system state is not confirmed prior to downloading.

**Failure to follow these instructions can result in death, serious injury or equipment damage.**

### Temporary Cycle Timing

The Workbench debugger can be used to temporarily change to ***Cycle Time Scanning*** 18 by writing a new value in the Cycle Timing (ms) entry field in the Target Execution toolbar. The current setting for the cycle time is also shown there (see below).


*Target Execution Toolbar*

## ⚠WARNING

**UNINTENDED EQUIPMENT OPERATION**

Evaluate the operational state of the equipment monitored and controlled by the RTU before debugging. Hazardous situations can occur if system state is not confirmed prior to debugging.

**Failure to follow these instructions can result in death, serious injury or equipment damage.**

The cycle time can also be set by in the debugger by writing to the ***__SYSVA_TCYCYCTIME*** 18 Resource *Global Variable*. The value entered uses the format of a TIME expression, e.g. T#200ms. Information on the Time Data type is available in the Workbench Help documentation.

To change the application scanning mode from cycle timing back to continuous scanning, set the Cycle Timing (ms) entry field back to zero or the **__SYSVA_TCYCYCTIME** variable value to "T#0s".

If the IEC 61131-3 application or the SCADAPack E RTU is restarted, a temporary cycle timing setting is lost. The cycle time will revert to the Resource properties' *Execution Mode* and *Cycle Time* settings.

## 5.3 High Priority Scanning

The SCADAPack E RTU supports increasing the priority of an IEC 61131-3 target VM task to achieve improved accuracy in the application cycle timing.

This is achieved by setting the “ISA_TASK_PRI” parameter using the *RTUPARAM* function block within the user application.

An IEC 61131-3 application can raise its own priority, subject to the following criteria:

- The Resource was built with Cycle Timing scanning (i.e. *Trigger Cycles* = TRUE and a non-zero *Cycle Time*). (See Section ***Cycle Time Scanning*** 18 above), and
- Each IEC 61131-3 application cycle needs to scan in less time than the fixed *Cycle Timing* setting.

High priority scanning sets the target VM task to run at a higher priority than routine RTU operating system tasks. This is useful where highly accurate execution rates are required.

An IEC 61131-3 VM task executing with standard priority and Cycle Time Scanning will still achieve relatively accurate execution timing, suitable for typical applications. Schneider Electric recommends that High Priority Scanning only be used where absolutely necessary for the application as it can adversely affect the RTU's capacity to perform other tasks.

If multiple IEC 61131-3 resources are executing on the same RTU, it is **not recommended** for more than one of the Resources to operate in high priority scanning mode.

Where high priority scanning is required, it is suggested that user IEC 61131-3 applications be split into a high priority and low priority component. This is to minimize the amount of application code that runs at high priority and the otherwise resultant impact on other RTU operations, such as communication. Two IEC 61131-3 target resources could be used to separately execute the low and high priority components. One of the resources could then use the RTUPARAM function block to increase its priority.

For more information on the RTUPARAM function block see the *SCADAPack E Target 5 Function Block Reference* manual.

# 6 RTU Logic Functionality

## 6.1 Input Scanning

### Binary Inputs

The update of binary input states to an IEC 61131-3 application, either from RTU local I/O, remote I/O or other RTU database data, is synchronized with the scanning of the IEC 61131-3 application resource. The IEC 61131-3 Resource updates its input I/O Devices at the start of the scan cycle as described in Section ***Target Scanning Cycle*** 14. The input variables that are updated from the I/O remain constant for the duration of one program scan. Physical RTU Digital inputs are represented as Binary Input Points in the RTU address space and are represented as *IEC BOOL* variables in an IEC 61131-3 resource.

### Analog Inputs

The update of analog input values to an IEC 61131-3 application is similar to binary inputs. The updates are synchronized with the scanning of the IEC 61131-3 application. The SCADAPack E RTU's database supports 16-bit Signed Integer, 32-bit Signed Integer as well as 32-bit Floating-Point data points. Analog variables in Workbench are represented as 16-bit Signed *IEC INT*, 32-bit Signed *IEC DINT* or 32-bit Floating Point *IEC REAL* within the application resource. The RTU supports data conversion between the 16-bit and 32-bit analog data types.

## 6.2 Output Updates

### Binary Outputs

RTU binary output data is updated at the end of the scan cycle following changes made by the IEC 61131-3 resource to its BOOL variables. See ***Target Scanning Cycle*** 14 for details. RTU Digital outputs and RTU derived database points are represented as Binary Points in the RTU address space and *IEC BOOL* variables in an application resource.

By default, an executing application with output variables connected to the physical I/O Devices has control of Physical RTU Binary Outputs and Derived RTU points, unless a “Remote Interlock” is active for individual binary points. See ***Remote Control Interlock*** 64 for more information.

See ***Stopping an IEC 61131-3 Application*** 52 for a description of the effect on Binary Outputs when the application resource is stopped.

### Analog Outputs

The analog output data is updated at the end of a Resource scan similar to Binary Outputs. The RTU supports 16-bit Signed Integer, 32-bit Signed Integer and Floating Point data objects. Analog variables are represented as16-bit Signed *IEC INT*, 32-bit Signed *IEC DINT* or 32-bit Floating Point *IEC REAL* within the Workbench environment. The RTU also supports data conversion between the 16-bit and 32-bit analog data types.

By default an executing IEC 61131-3 application, with analog points on Output I/O Devices, has control of those Physical RTU Analog Outputs and Derived RTU points, unless a “Remote Interlock” is active for individual analog points. See ***Remote Control Interlock*** 64 for more information.

See ***Stopping an IEC 61131-3 Application*** 52 for a description of the effect on Analog Outputs when the application resource is stopped.

## 6.3 Language Types

### Language Types

Projects can be developed using different programming languages from the IEC 61131-3 standard. When building, resources are compiled to produce very fast "target independent code" (TIC) code.

The SCADAPack Workbench supports four IEC 61131-3 international standard sequencing languages. These are:

- SFC - Sequential Function Chart
- FBD - Function Block Diagram
- LD - Ladder Diagram
- ST - Structured Text

Control applications may be written in SCADAPack Workbench using any combination of the above languages, and can be executed on either, or both IEC 61131-3 target resources. The *Scientific Apparatus Makers Association* (SAMA) diagram language and the IEC 61499 language are **not** supported.

## 6.4 Data Types

The SCADAPack E RTU supports the following IEC 61131-3 data types and corresponding Point values.

**Relationship between IEC 61131-3 and SCADAPack E Data**

| Workbench Data Type | Data Format | Use IEC61131-3 interfaces | RTU Database point representation |
|---|---|---|---|
| BOOL | TRUE or FALSE | *RTU_BIN_* I/O devices, *GETPNTB*, *SETPNTB* function blocks | Binary point state |
| INT | Signed, short integer (16-bit) | *GETPNTSS*, *SETPNTSS* function blocks | Analog point, 16-bit Integer value (*RAW* |
| DINT | Signed, long integer (32-bit) | *RTU_RAW_* I/O Devices, *GETPNTSL, SETPNTSL* function blocks | Analog point, 32-bit Integer value (*RAW* |
| UINT | Unsigned, short integer (16-bit) | *GETPNTUS*, *SETPNTUS* function blocks | Analog point, 16-bit Integer value (*RAW* ) |
| UDINT | Unsigned, long integer (32-bit) | *RTU_COUNTER_READ* I/O device, *GETPNTC* function block | Counter point, 32-bit Integer value |
| REAL | Floating point 32-bit IEEE-754 | *RTU_ENG_* I/O devices, *GETPNTF*, *SETPNTF* function blocks | Analog point, 32-bit floating point Engineering value (*ENG*) |
| TIME | 1 ms counts | *LOC_TIME* function block | |
| STRING | Up to 255 characters | *RTU_STRING_WRITE* I/O device, *RDSTRING* function block | System string point |

The target on the SCADAPack E RTU supports elementary IEC 61131-3 data types, with the exception of the SAFEBOOL type. See the Workbench Help documentation for more information on the IEC 61131-3 data types.

Workbench has a wide range of built-in Data Conversion operators such as ANY_TO_DINT, ANY_TO_TIME, etc.

IEC 61131-3 integer variables can correspond to RTU analog point Integer (RAW) values when connected through I/O Devices or function blocks. REAL IEC61131-3 variables correspond to RTU analog point Engineering (ENG) values when connected through I/O Devices or Function Blocks.

IEC 61131-3 Integer constant data types, such as DINT, INT, UDINT, etc. may be expressed with one of the following Bases. Integer constants begin with a Prefix that identifies the Bases used:

| Base | Prefix | Example |
|---|---|---|
| DECIMAL | (none) | -260 |
| HEXADECIMAL | "16#" | 16#FEFC |
| OCTAL | "8#" | 8#177374 |
| BINARY | "2#" | 2#0101_0101_0101_0101 |

The underscore character ('_') may be used to separate groups of digits. It has no particular significance other than to improve literal expression readability.

## 6.5 Operators

SCADAPack Workbench application software supports a comprehensive array of built-in operators including functions for:

**Table 5.2: Workbench Supported Built in Operators and Functions**

| | |
|---|---|
| boolean operations | AND, OR, XOR, NOT, F_TRIG, R_TRIG, SR, RS |
| analog operations | *, +, -, /, AND_MASK, CMP, LIMIT, MAX, MIN, MOD, NEG, NOT_MASK, ODD, OR_MASK, RAND, ROL, ROR, SHL, SHR, STACKINT, XOR_MASK |
| data manipulation | 1 GAIN, MUX4, MUX8, SEL |
| type conversion | ANY_TO_xxx (18 data types), CHAR |
| comparison | <, <=, <>, =, >, >= |
| maths | ABS, ACOS, ASIN, ATAN, COS, EXPT, LOG, POW, SIN, SQRT, TAN, TRUNC |
| string management | ASCII, DELETE, FIND, INSERT, LEFT, MID, MLEN, REPLACE, RIGHT |
| timer control | TOF, TON, TP |
| control/signal handling | AVERAGE, BLINK, DERIVATE, INTEGRAL, HYSTER, LIM_ALRM, SIG_GEN |
| Counting | CTD, CTU, CTUD |
| Date | CURRENT_ISA_DATE, SUB_DATE_DATE |

See the Workbench Help documentation sections on Operators, Functions & Function Blocks for more information.

## 6.6 I/O Locking

I/O locking provides the ability to lock inputs and outputs into a certain state, regardless of their true state. This provides a mechanism for freezing I/O in a fixed state, allowing maintenance to be performed while the IEC 61131-3 application is still executing.

**WARNING**

**UNEXPECTED EQUIPMENT OPERATION**

Evaluate the operational state of the equipment monitored and controlled by the SCADAPack E RTU prior to initializing the SCADAPack E RTU.

**Failure to follow these instructions can result in death, serious injury or equipment damage.**

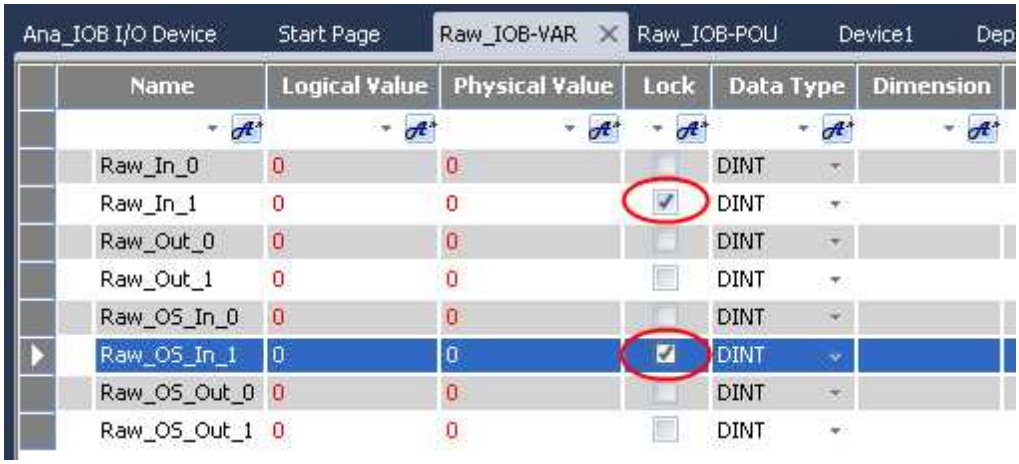
SCADAPack 300E RTU's "**FORCE**" LED indicates that an application resource has at least one I/O variable in a LOCK state. When no variables remain locked, the FORCE LED will turn off.

I/O variables on the target can be locked and unlocked individually through the SCADAPack Workbench when connected online with the Workbench debugger to the SCADAPack E RTU's target resource. This can be done through the resource's Dictionary dialog (shown below) or from the *I/O Device* Window.

When an I/O variable is locked, the Workbench indicates the Logical Value used by the IEC 61131-3 Resource and the Physical Value associated with the I/O Device connection.

- For an input variable (Direction *VarInput*), the Logical Value field can be modified to override the variable value used by the Resource. The Physical Value indicates the actual value coming from the I/O Device even though it is not being used by the Resource.
- For an output variable (Direction *VarOutput*), the Physical Value can be modified to override the value output to the I/O Device. The Logical Value indicates the variable value coming from the Resource even though it is not being sent to the I/O Device.

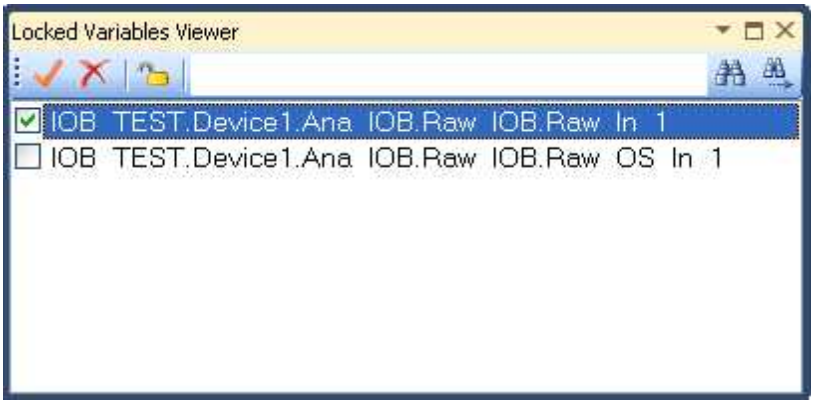
Clear the Lock indication to release the override on the variable. The Physical Value and Logical Value should return to indicate the same value.




*Variable locking in the Resource Dictionary dialog*

The Workbench debugger also provides another mechanism to view or unlock locked variables. When in

Debug mode, use the Lock Variables Viewer (shown below). This can be activated from the menu **Debug > Locked Variables Viewer** or from the lock icon on the Target Execution Toolbar.



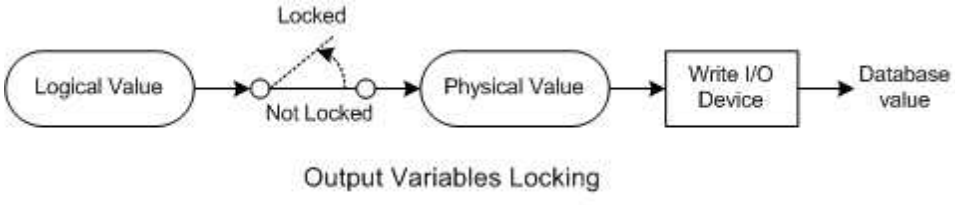

*Locked Variables Viewer Dialog*

## ⚠WARNING

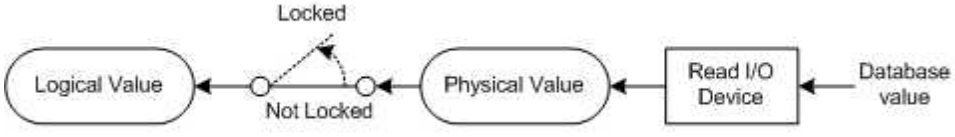
**Online firmware update will NOT preserve the RTU operational state prior to the upgrade, including output states, application state information, I/O Lock states, etc.**

**Consider the impact on operating equipment when upgrading firmware.**

The diagram below shows how inputs and outputs are isolated from the IEC 61131-3 variable database and Workbench debugger when locked.




Output Variables Locking




Input Variables Locking

## 6.7 String Variables & Conversion

### String Variables

STRING variables provide a mechanism for manipulating groups of ASCII characters (strings). For example, ASCII characters can be read in from external equipment, processed by an IEC 61131-3 program, which may then build and output an appropriate ASCII reply.

STRING variables also interface with the SCADAPack E RTU system string points. SCADAPack E IEC 61131-3 string variables are limited to a max. 255 characters in length.

### Converting a String to Integer Values

The built-in IEC 61131-3 operators **ANY_TO_xxx** (18 types) can convert other IEC data types to almost any other type. When **ANY_TO_DINT** is used to convert a STRING data type to a DINT integer, IEC 61131-3 does not automatically perform bounds checking on the input message value. Where a STRING message string has a value greater than 2^31, unexpected results may occur.

It is recommended that the user perform their own bounds checking on the message value if it is required to handle numbers above 2^31. For example, use **ANY_TO_LINT** and then check the value.

## 6.8 Retained Variables

The SCADAPack E RTU provides a facility for storing IEC 61131-3 application resource variable values that are required to retain their value when power is lost, or when an application resource is restarted. A copy of every application variable type with the Retain attribute checked is stored by the RTU, at the end of every Resource scan (for the two target resources), in non-volatile areas of RTU memory. After a user application restarts, retained variables for that application are refreshed with their last saved value.

**Variables using a direction of VarOutput or VarInput and Retained enabled are not supported. It is possible to select this in some versions of SCADAPack Workbench. However, the variable will not be retained. Do not use these combinations.**

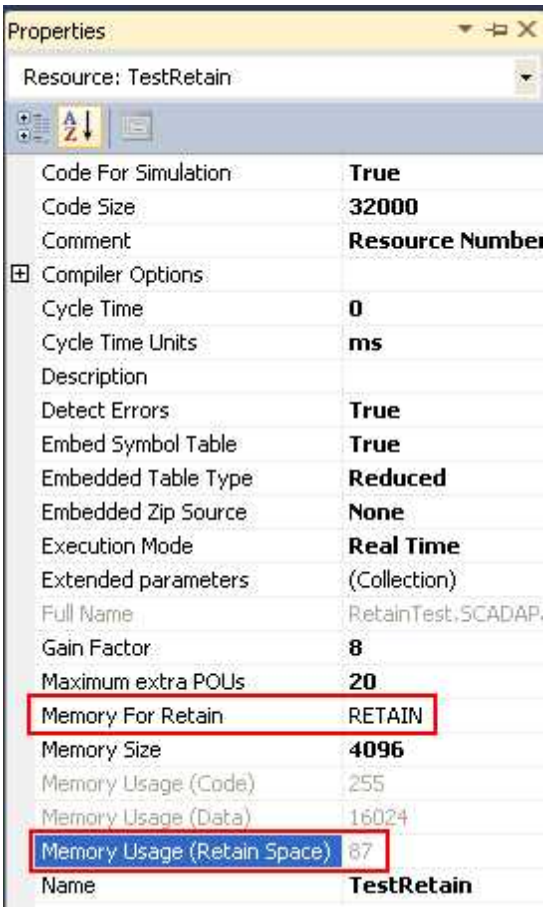
4K of RTU system NV-RAM (per-target resource) is set aside for retained variables, which is shared by the standard variable types. The table below shows the memory used for each of the standard types.

**Example memory used by standard IEC 61131-3 variable types**

| IEC 61131-3 Variable Type | RTU Memory Usage |
|---|---|
| BOOL | 1 byte |
| INT / UINT | 2 bytes |
| DINT / UDINT | 4 bytes |
| TIME | 4 bytes |
| STRING | 255 bytes |

The amount of memory allocated to each variable type is automatically allocated by the SCADAPack E RTU depending on the application requirements and the number of active applications in the RTU. The SCADAPack Workbench resource *Memory For Retain* property is **not** required for the SCADAPack E to use *retained variables*. Non-volatile memory storage for retained variables is allocated at a fixed memory segment within the RTU. The total retained memory space requirements of a resource needs to be less than 4KB or a Resource startup status code will be set (Code 4004).

The *Memory Usage (Retain Space)* property of each resource shows the memory (in bytes) requirement for the application.




*Workbench Resource Properties window*

The SCADAPack E RTU clears retained variable values when an IEC 61131-3 application or RTU NV-RAM is cleared.

E.g. when a Cold Boot factory-defaults initialization occurs, if RTU firmware is changed through a local, manual firmware upgrade, or when using the command line "CLEAR ISaGRAF" command.

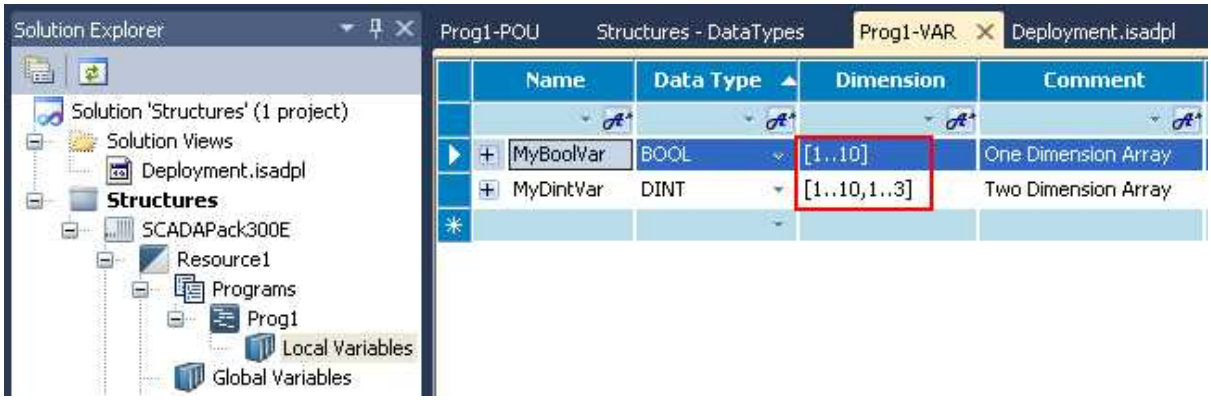
The SCADAPack E IEC 61131-3 targets in the RTU automatically clear retained variables for the appropriate target kernel task when a new IEC 61131-3 application is loaded.

## 6.9 Array Variables

### Static Arrays

The SCADAPack Workbench supports arrays of standard IEC 61131-3 types or user defined structured types. An array has one or more dimensions. Array dimensions are positive DINT literal expressions and array indexes are DINT literal expressions or variables.

Array names can have up to 128 characters and can begin with letters or single underscores followed by letters, digits, and single underscores. Array variables are declared in a Resource's Global Variables window or in a Program's Local Variables window.




*Declaring Arrays in the Local Variables Window*

Examples:

**1. One-dimensional array:**

MyBoolVar is an array of 10 BOOL. Its dimension is defined as follows: [1..10].

Structured Text example initialization:

```
FOR i := 1 TO 10 DO

  MyBoolVar [i] := FALSE;

END_FOR;
```

**2. Two-dimensional array:**

MyDintArray is an array of DINT. It has two dimensions defined as follows: [1..10,1..3]

Structured Text example initialization:

```
FOR I := 1 TO 10 DO

  FOR J := 1 TO 3 DO

    MyDintArray [I][J] := FALSE;

  END_FOR;
```

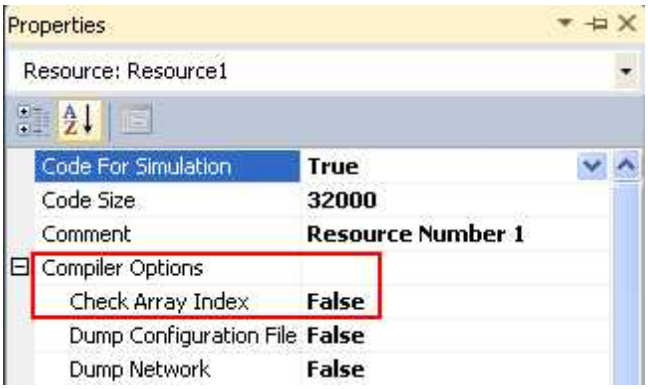
```
END_FOR;
```

Array access example:

```
MyDintArray [1,2] := 100;
```

| CAUTION |
| --- |
| **Like many other Programming Languages, access of an IEC 61131-3 array variable in a program using an index number beyond the configured dimension can lead to serious unpredictable results including overwriting of data, reset of the RTU, etc.** |

The Resource *Compiler Options* property, *Check Array Index* can be enabled to check the bounds on array accesses at run-time (see below). If an out-of-bounds access is found, System Error 4033, "Kernel TIC: Boundary check error" is set and the offending Resource is stopped.




*'Check Array Index' Property*

| CAUTION |
| --- |
| **Enabling the Check Array Index Compiler Option should not be relied upon to detect all array access errors. If enabled, this option is likely to have a performance impact on the application's cycle time.** |

Once an array is defined in a resource's Local or Global Variables Window, the IEC notation ArrayName [ index ] may be used in any program or I/O Device connection.

When using IEC 61131-3 arrays, keep in mind:

- Functions and Function Blocks cannot have parameters of array type, but can have an element of the exact element ( e.g. using notation ArrayName [7] )
- Array elements starts from element index 0 or 1, depending on how it was declared in the resource's Dictionary.
- Array elements can be initialized individually in the resource's Dictionary. Or, write initial values using a FOR loop in a startup program.

- The *Retained* property applies to the whole array. It's not possible to retain just individual elements.

## Dynamic Arrays

The SCADAPack Workbench allows up to 16 arrays of DINT values to be created dynamically (per-resource) using the ARCREATE Function. This feature may be useful in user functions where the bounds of an array is determined by a parameter where the value is not known at compile time.

The dynamic array is created with a single call to the ARCREATE function, passing in an array ID value between 0 and 15. This ID value is used in subsequent calls to the ARREAD (read from array) and ARWRITE (write to array).

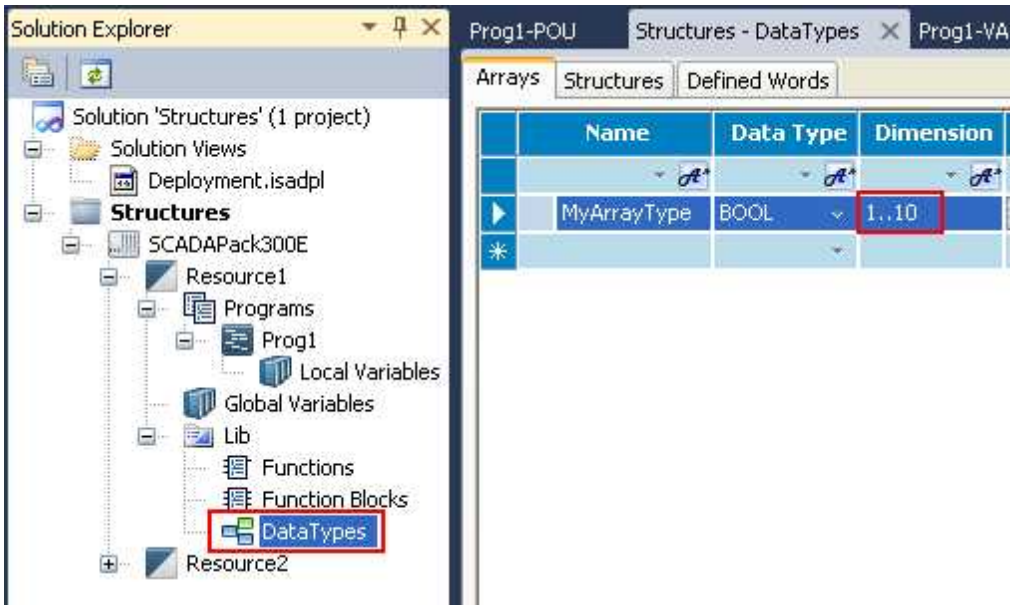
Dynamic arrays are cleared when the application resource is stopped or the RTU is restarted. The array element values cannot be retained in dynamic arrays.

See the *SCADAPack E Target 5 Function Blocks Reference* manual for further details.

### 6.9.1 Array Data Types

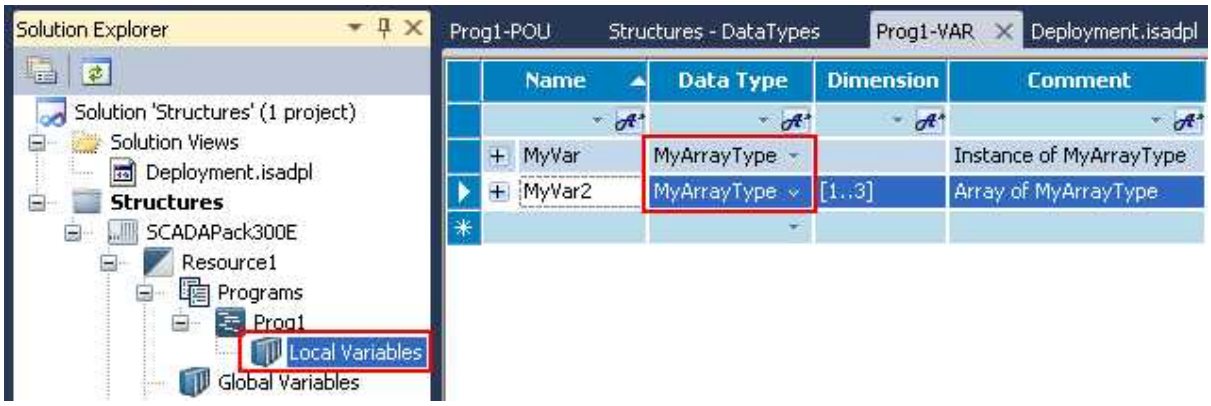
SCADAPack Workbench also supports the concept of custom Array Data Types. The user can create their own named array type and then add variables of this type in the Resource's *Global Variables* or Program's *Local Variables* window.

Array Data Types are declared in the Resource's **Lib\DataTypes** Window as shown below.




*Declaring an Array Data Type*

In order to use a custom Data Type, add variables of that type using the Resource *Global Variables* window or Program *Local Variables* window.




*Adding variables of Array Data Types*

Example 1:

The user creates an Array Data Type called *MyArrayType*, which is an array of 10 BOOL. Its dimension is defined as follows: [1..10].

In the Resource's Local Variables dictionary, the array *MyVar* is created, which is of type *MyArrayType*. The *MyVar* array can then be used as a normal array in a program as follows:

```
Ok := MyVar [4];
```

In this example, the *MyVar* variable is an *instance* of the *MyArrayType* Data Type.

Example 2:

In the Resource's Local Variables dictionary, the array *MyVar2* is created, which is an array of type *MyArrayType* (as per the previous example). This effectively creates a two-dimension array, which could be initialized as follows:

```
FOR I := 1 TO 3 DO

  FOR J := 1 TO 10 DO

    MyVar2 [I][J] := FALSE;

  END_FOR;

END_FOR;
```

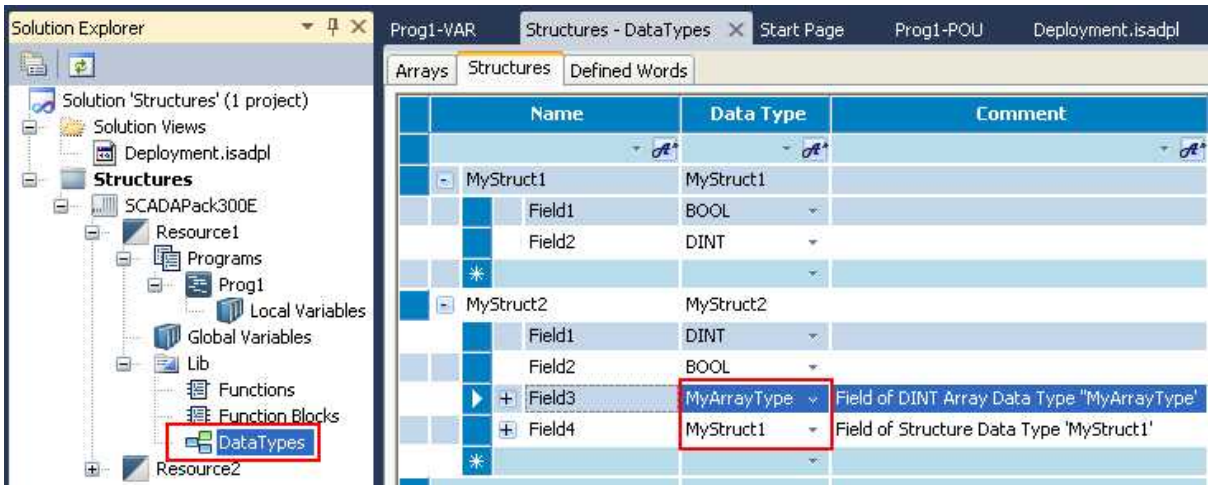
## 6.10 Structure Data Types

SCADAPack Workbench also supports the concept of Structure Data Types made up of elementary IEC 61131-3 types or user defined structured types.

A structure is composed of sub-entries called Fields. The user can create their own named structure type and then then define variables of this type in the Resource's *Global Variables* or Program's *Local Variables* window.

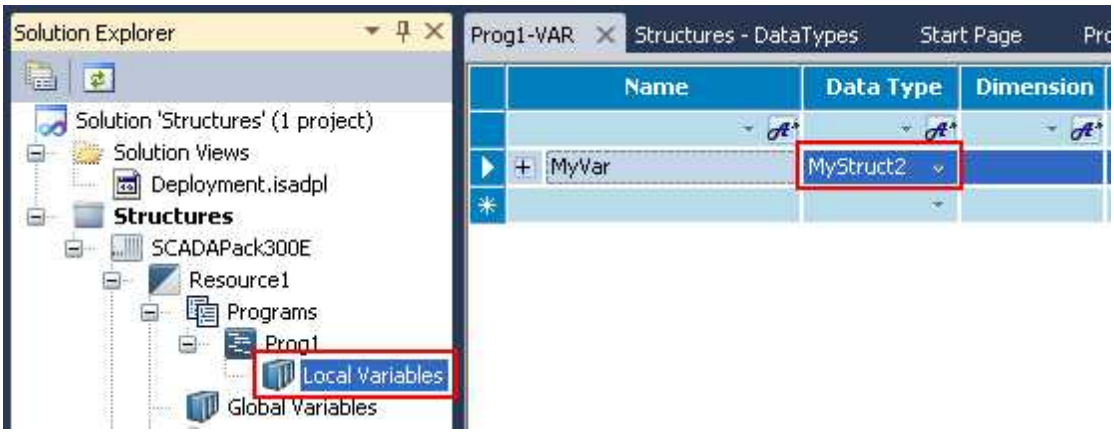
Structure names can have up to 128 characters and can begin with letters or single underscores followed by letters, digits, and single underscores.

Structure Data Types are declared in the Resource's **Lib\DataTypes** Window as shown below.




*Declaring Structure Data Types*

In order to use a Structured Data Type, add variables of that type using the Resource *Global Variables* window or Program *Local Variables* window.




*Declaring instances of Structure Data Types*

Example:

In the Data Types *Structure* tab, the user creates *MyStruct1*, which is composed as follows:

Field1 which is BOOL

Field2 which is DINT

A second Structure, *MyStruct2* is also created as follows:

Field1 which is DINT

Field2 which is BOOL

Field3 which is of ***Array Data Type*** 37 *MyArrayType* (an array of 10 DINT)

Field4 which is of type *MyStruct1*

In the Resource's Local Variables dictionary, the array *MyVar* is created, which is of type *MyStruct2*. The following Structured Text example shows how the *MyVar* variable could be used in an IEC 61131-3 program:

```
Value1 := MyVar.Field1;      (* Value1 is of type DINT *)

Ok1 := MyVar.Field2;         (* Ok1 is of type BOOL *)

Tab[2] := MyVar.Field3 [5]; (* Tab is an array of DINT *)

Value2 := MyVar.Field3 [8]; (* Value2 is of type DINT *)

Ok2 := MyVar.Field4.Field1; (* Ok2 is of type BOOL *)
```

- Structure fields can be initialized individually in the Workbench Dictionary. Or, write initial values in a startup program.
- The *Retained* property applies to the whole structure. It's not possible to retain just individual fields.
- Functions and Function Blocks cannot have parameters of structure type, but can have an element of the exact field, if it's an elementary IEC 61131-3 type ( e.g. using 'dot' notation MyVar.Field1 ).

## 6.11 Second IEC 61131-3 Resource Features

A user IEC 61131-3 application using the RTU's second target resource may connect to any RTU points on Input I/O Devices, but cannot be connected to output I/O devices with points that are already on an output I/O device used by the other Resource.

The Workbench Debugger will dynamically switch communication between two Resources running on the target, as requests are received to/from each.

The SCADAPack Workbench Resource Numbers are configured as "1" and "2" to represent the RTU's first and second target resources, respectively.

## 6.12 Defined Words

The SCADAPack Workbench enables the re-definition of literal expressions, true and false Boolean expressions, keywords or complex ST expressions. To achieve this, an identifier name, called a Defined Word, is assigned to the corresponding expression. Defined words have a **global scope**, i.e., they are available for use in any component of an application resource. The following are examples of defined words:

YES is TRUE

PI is 3.14159

OK is (auto_mode AND NOT (alarm))

When such an equivalence is defined, its identifier is available anywhere in an ST program to replace the attached expression. The following ST programming example uses defined words:

```
If OK Then
  angle := PI / 2.0;
  isdone := YES;
End_if;
```

When the same identifier is defined twice with different ST equivalencies, the last defined expression is used:

Define:

OPEN is FALSE

OPEN is TRUE

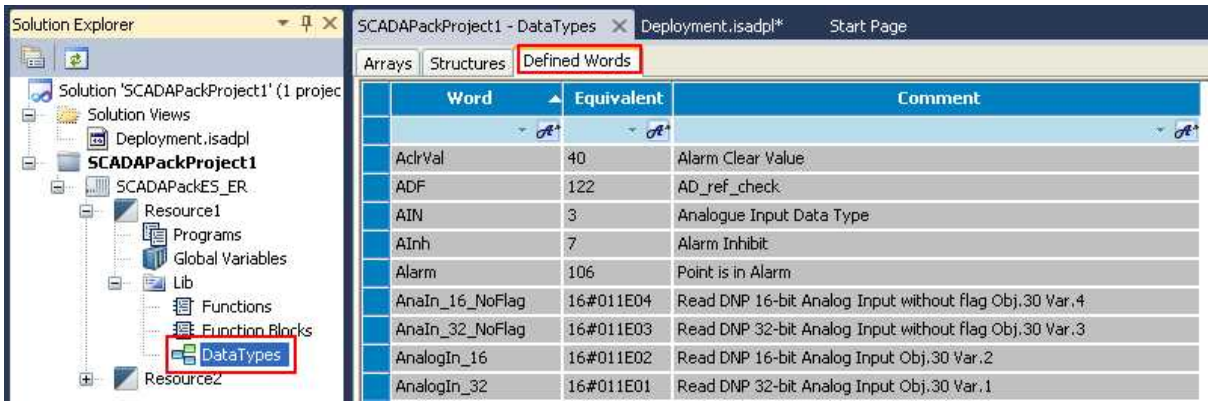
means:

OPEN is TRUE

Names of defined words can have up to 32 characters and begin with a letter followed by letters, digits, and single underscore characters. The last character can be either a letter or a digit.

The definition of a defined word is expressed using variables or literals; these cannot use defined words or expressions.

Defined Words are declared in the SCADAPack Workbench Resource's **Lib\DataTypes** Window as shown below.




*Accessing Defined Words in a project*

The 'greyed-out' Defined Words shown in the above diagram are target-specific. They are defined by the SCADAPack Workbench and can't be removed. However, user-specific Defined Words can be added to the list.

The list of Defined Words can be alphabetically and numerically sorted with the grid column headers.

# 7 Application Management

The following sections describes facilities provided by SCADAPack Workbench for application management

## 7.1 Online Changes

Online changes allows an IEC 61131-3 application resource running on the SCADAPack E RTU to be modified, without any discontinuity of execution. Use of this feature is subject to available RAM memory on the RTU and similarity of the new application with the executing application.

| ⚠WARNING |
| --- |
| Online changes should be used with care. The IEC 61131-3 target may not detect all possible conflicts generated by user-defined operations as a result of these online changes. |

An Online Change does not restart the resource. Variable values are preserved. When performing online changes **you cannot** modify application parameters, add or remove I/O Devices, modify I/O Device connections or modify I/O Device parameters.

The following application modifications **are permitted** when performing Online Changes:

| | |
| --- | --- |
| Internal Variables | When renaming or changing the data type of internal variables, the Workbench creates new variables. Therefore, variables are initialized.<br><br>Adding and removing elements in arrays for internal variables. For multi-dimensional arrays, you can only add elements to the first dimension. The Workbench initializes these new elements. Adding elements to other dimensions causes the Workbench to initialize a new array. |
| Programs | Adding, deleting, renaming, and reordering (for execution within the programs section) programs. When renaming programs, the Workbench detects a CRC mismatch and updates the code on the target for the program and reinitialize's local variables. When renaming SFC programs, instance data and local variables are not preserved, i.e, elements are reset to their initial state.<br><br>When planning to add programs (other than SFC) using online changes, you need to allocate a sufficient number of maximum extra POUs.<br><br>When planning to add SFC programs using online changes, you need to allocate sufficient memory space for SFC programs.<br><br>Adding, deleting, renaming steps and transitions as well as modifying the initial step or the flow between elements. When modifying SFC programs, instance data and local variables is preserved, i.e., elements are not reset to their initial state.<br><br>Adding, deleting, and moving action blocks within steps of SFC programs. Action blocks within steps are executed in the order of appearance. You can also change the qualifier of an action block. |
| Functions and Function Blocks | Adding, deleting, and moving function blocks. Adding and deleting function block instances.<br><br>Renaming and modifying user-defined functions and function blocks. |

| | |
|---|---|
| | Adding, removing, and modifying the parameters of user-defined functions and function blocks. When modifying the parameters, instance data is not preserved. You need to recompile modified functions and function blocks called by other POUs as well as the calling POUs. |
| I/O Devices | New I/O variables may be created and attached to an existing I/O Device.<br>Existing I/O variables may be moved from channel to channel on the same I/O Device.<br>Existing I/O variables may be moved from one I/O Device to another.<br>Parameters may be modified on an existing I/O Device except for the following:<br>• Parameters cannot be swapped between I/O devices. This is because the validation checks are done one device at a time; the first device to be checked will detect a duplicate and refuse the change (because the other device hasn't been changed yet).<br>• The 'Port' parameter cannot be changed online unless it's a TCP port number.<br>If an I/O output device has channels connected to local physical outputs and the 'First_Point_Number' parameter is changed online, the device will follow the state of the 'Hold_On_Stop' parameter (if applicable). If the 'Hold_On_Stop' parameter is False (default), the I/O device will zero the current value or state of any points that have been disconnected from output channels as a result of the online change. See the Hold On Stop 64 section.<br>The following actions require a full application download:<br>• Adding a new I/O Device<br>• Removing an existing I/O Device<br>• Modifying the number of channels on an existing I/O Device |

**⚠WARNING**

**UNCONTROLLED OUTPUT VARIABLES**

Online changes to output I/O devices should be done with care. If output variables are moved between channels or devices, the vacated channels will remain in their previous state or value, unless replaced with another variable.

**Failure to follow these instructions can result in death, serious injury or equipment damage.**

You can perform online changes after building a project. Online changes are unavailable after cleaning a project.

Modifying a running application with the Online Change feature consists of the following operations:

- Modify the application source code in the Workbench
- Build the current application program
- Download the new application using the Online Change command. From the Workbench Solution Explorer, right-click the project for which to perform the Online Change, then choose *Online Change* or

click the  button on the Workbench Toolbar.

**⚠WARNING**

**UNINTENDED EQUIPMENT OPERATION**

Evaluate the operational state of the equipment monitored and controlled by the RTU before downloading. Hazardous situations can occur if system state is not confirmed prior to downloading.

**Failure to follow these instructions can result in death, serious injury or equipment damage.**

- The user will be informed if the change cannot be made online, and the change will not proceed.

## 7.2 Application Storage

The IEC 61131-3 application(s) loaded into the SCADAPack E RTU is stored in a non-volatile file system directory on the RTU called "C:\ISaGRAF5". When power is restored after an outage, each stored application is validated by the RTU and a separate copy is made in volatile RAM for execution.

If both copies are not valid, the IEC 61131-3 target waits for a new application to be loaded and the *Application Halted* system binary point for that Resource is activated (RTU system binary point 50100 or 50101).

Variables, other than those specified in the resource as *retained variables*, are stored in volatile (non-saved) memory. The IEC 61131-3 target initializes their values when the application restarts.

The IEC 61131-3 applications are stored in the RTU file system in files called:

- "ISPxxx01" is the resource's compiled TIC code.
- "ISPxxx03" is the resource's configuration file.
- "ISPffe0d" is the Network configuration file (shared between resources)
- "IDSxxx01" is the resource's Symbol Table information.
- "IDSxxx03" is the resource's *Embedded Zip Source* data (if configured).

Where: "xxx" is the resource number, eg. "001".

The SCADAPack Workbench supports resource numbers "1" and "2" corresponding to the two SCADAPack E IEC 61131-3 target Resources.

## 7.3 Storing a Project on the RTU

SCADAPack Workbench provides the facility for storing the IEC61131-3 "source" on the target RTU along with the executing TIC code.

This facility allows the original code that generated the Workbench application to be stored on the SCADAPack E RTU, and later uploaded to the Workbench.

The application upload facility is supported through the following SCADAPack E IEC 61131-3 interfaces:

- via an RTU serial port configured as *ISaGRAF*
- via Ethernet when RTU has ISaGRAF/TCP service enabled (see SCADAPack E Configurator **TCP/IP** page)

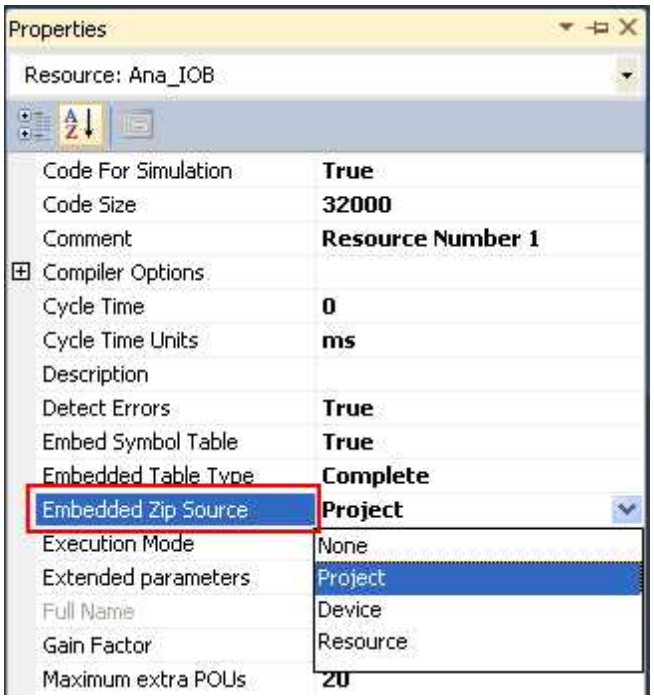
IEC 61131-3 project Upload facilities are **not** supported via DNP3 communications using SCADAPack E Configurator.

### Downloading a Workbench Project to the Target

The SCADAPack Workbench project "source" can be downloaded along with the RTU TIC code. The project is compressed by the Workbench and forms part of the application resource downloaded to the RTU.

This applies to applications downloaded via the Workbench Debugger or via file transfer in an ".I5P" package file. (See ***Remote Target Access*** 80).

'To include Workbench Project "source" with the execution code to download to the target, set the Resource property *Embedded Zip Source* settings as shown below.




*Embedding application source code*

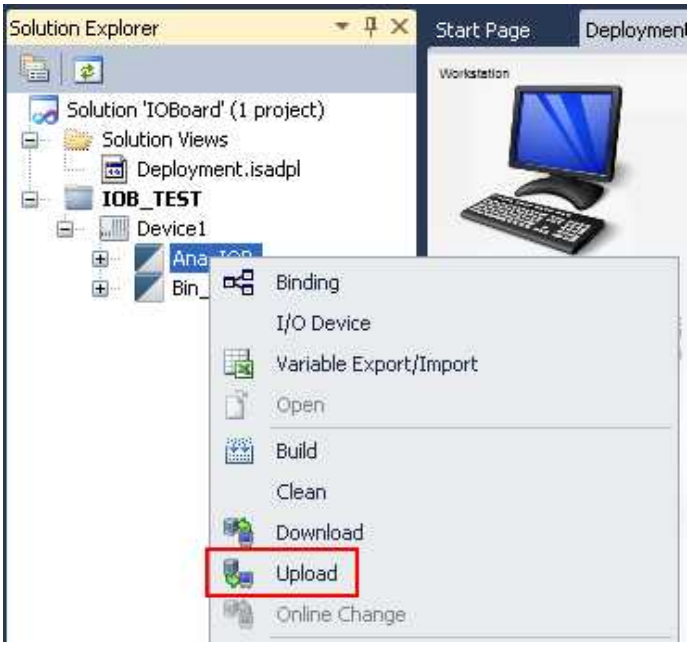
The user can choose to embed source for just the selected *Resource* or for the whole *Project* (i.e. both project resources).

When the application is next built by the Workbench and downloaded to the RTU, it will store the project on the RTU for later retrieval by ***Uploading*** 49.

## Uploading a Project from the Target to the Workbench

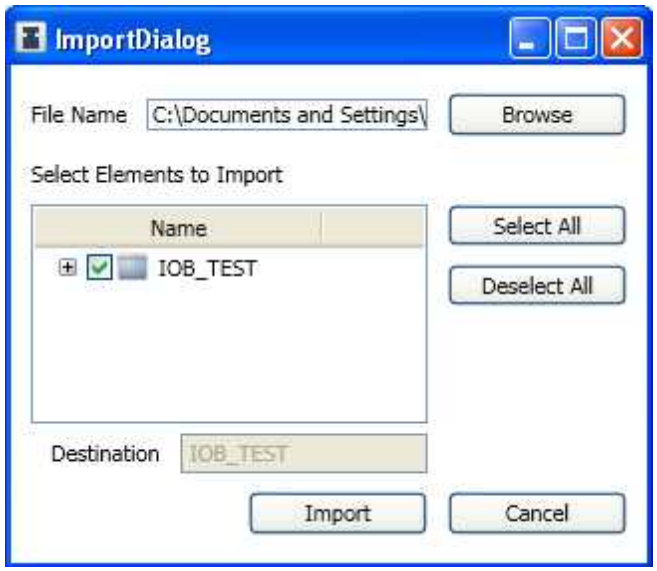
The SCADAPack Workbench allows a project's source to be retrieved from the IEC 61131-3 target if it was originally build with the *Embedded Zip Source* resource property.

In the Workbench Solution Explorer window, right click the desired resource to upload. It shows the dialog in the following figure:




*Select a Resource to Upload*

Select *Upload* and wait while the data is uploaded from the target. When this is complete, an Import dialog will be shown as follows:

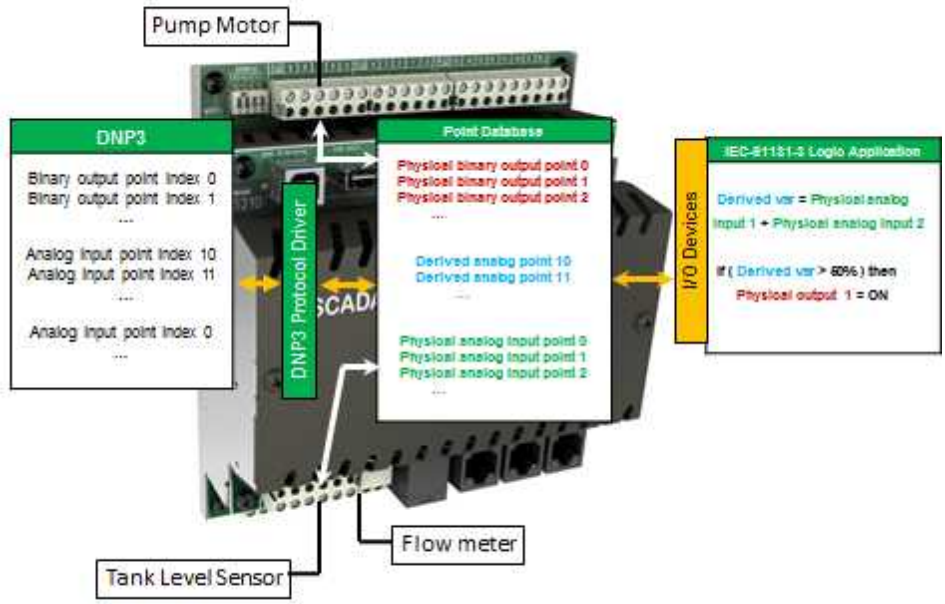



*Resource Import Dialog*

Selecting the *Import* button will load a copy of the uploaded resource into the Workbench Solution Explorer window.

# 8 I/O Interfaces

The diagram below shows the relationship between the IEC 61131-3 application and other elements that it interacts with in the SCADAPack E RTU firmware. The IEC 61131-3 application connects to the RTU point database via I/O Devices and Function Blocks. In turn, the point database segregates the I/O (physical & derived), the protocol drivers and the IEC 61131-3 application.




SCADAPack Workbench provides several interfaces for associating RTU data with IEC 61131-3 variables. The following sections describe the methods that can be used.

## 8.1 Physical I/O

Physical inputs and outputs on the SCADAPack E RTU can be accessed by the IEC 61131-3 resource by using I/O Devices. RTU internal data points may also be accessed via I/O Devices (or via other Function Blocks) as will be described later in this section. Each I/O Device needs to be supplied with a Point Number that specifies the RTU starting point index or offset when reading from inputs or writing to outputs. This point number is entered into the ***first_point_number*** field of the particular IEC 61131-3 I/O device using the Workbench Resource's *I/O Device* window.

For information on the SCADAPack E RTU physical point index mapping see the following manuals, where relevant:

- *SCADAPack E 5000 Series I/O Expansion Reference* manual
- *SCADAPack E Configurator User Manual*
- *SCADAPack E DNP3 Technical Reference* manual

### Stopping an IEC 61131-3 Resource

The effect of stopping an IEC 61131-3 Resource on Physical I/O varies between SCADAPack E Smart RTU.

The following table defines the effects.

| When ISaGRAF application is Stopped: | SCADAPack ES local I/O | SCADAPack ER I/O cards | SCADAPack 300E I/O | SCADAPack ES Remote I/O | 5000 Series I/O | Remote RTU & IED Devices * |
|---|---|---|---|---|---|---|
| Binary Inputs updated in database? | Yes | Yes | Yes | Yes | Yes | Yes |
| Analog Inputs updated in database? | Yes | Yes | Yes | Yes | Yes | Yes |
| Binary Output & database state if I/O device not using "Hold_On_Stop" parameter | Reset Off | Reset Off | Unchanged | Unchanged | Unchanged | Unchanged |
| Binary Output & database state if I/O device is using "Hold_On_Stop" parameter | Unchanged | Unchanged | Unchanged | Unchanged | Unchanged | Unchanged |
| Analog Output & database value if I/O device is not using "Hold_On_Stop" parameter | Reset to 0 | Reset to 0 | Unchanged | Unchanged | Unchanged | Unchanged |
| Analog Output & database value if I/O device is using "Hold_On_Stop" parameter | Unchanged | Unchanged | Unchanged | Unchanged | Unchanged | Unchanged |
| Binary Output changes if database value updated after Resource Stop | Yes | Yes | Yes | Yes | Yes | Yes |
| Analog Output changes if | Yes | Yes | Yes | Yes | Yes | Yes |

| | | | | | | |
|---|---|---|---|---|---|---|
| database value updated after Resource Stop | | | | | | |

*Applies to DNP3 Master and IEC60870-5-103 Master communications to remote devices

See ***Hold On Stop*** 64 for output I/O Device parameter information.

The above table does not include data to/from PLC devices such as Modbus, DF1, etc. The IEC 61131-3 application manages the data transfer for PLC protocol communications and so when a resource is stopped, PLC communications and data transfer ceases.

## 8.2 Derived Data

Derived RTU data (e.g. calculations to be sent to a SCADA master, data for peer-to-peer communications, etc.) can be accessed by a IEC 61131-3 Resource in exactly the same way as physical I/O, i.e. using the I/O device mechanism (or function block mechanism). Each I/O device type used to access physical I/O may be used for manipulating derived RTU data, by simply specifying an appropriate ***first_point_number***. Derived RTU points can be created using SCADAPack E Configurator. Once created, a user IEC 61131-3 application can open I/O devices attached to the derived points.

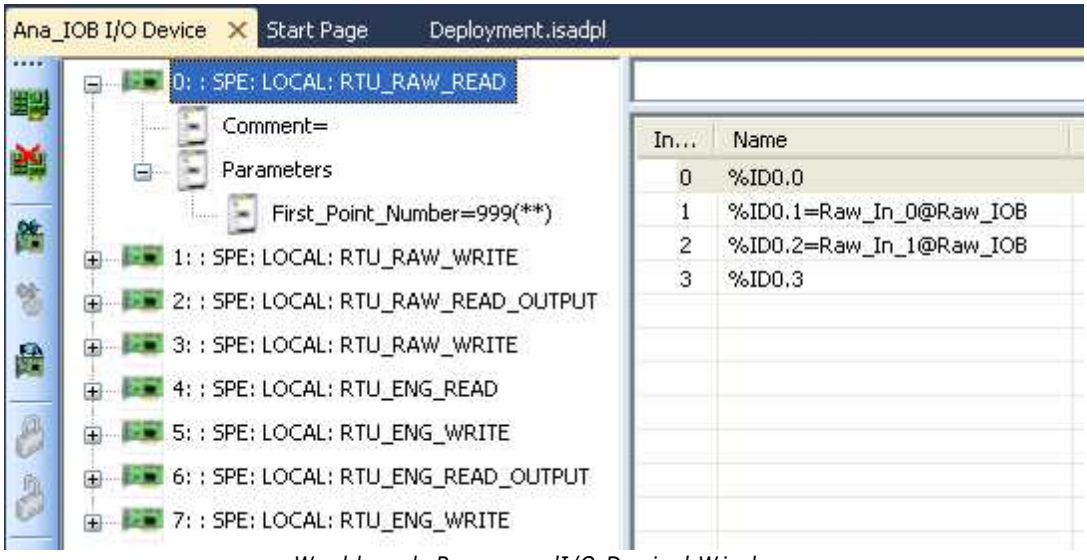
Analog I/O devices can receive data from and send data to an IEC 61131-3 Resource in integer or real (floating point) format from derived or physical I/O points (e.g. for SCADA). Floating point RTU data is related to integer data through Raw and Engineering scaling parameters configured for each RTU database point.

SCADAPack E **RTU_xxx_WRITE** I/O devices *export data* from a IEC 61131-3 Resource to the RTU database. This data is then available to the SCADA master system or other peer RTUs. Similarly, **RTU_xxx_READ** I/O devices used for derived points *import data* from the RTU database into a IEC 61131-3 Resource. Control data from a SCADA master, or peer RTU for example is then available to the IEC 61131-3 Resource.

IEC 61131-3 Resource derived outputs = SCADA master inputs, Peer RTU input data

IEC 61131-3 Resource derived inputs = SCADA master outputs, Peer RTU output data

IEC 61131-3 input variables attached to RTU derived database point (via I/O Device channels) may be initialized to a predefined state via the appropriate **SETPNTxx** Function Block. The figure below shows an example of the Workbench I/O Device interface, with inputs attached to the currently selected device (RTU_RAW_READ) displayed on the right.




*Workbench Resource 'I/O Device' Window*

## 8.3 I/O Devices

The list below contains the "RTU" I/O Devices available in the Schneider Electric SCADAPack Workbench for the SCADAPack E RTU.

The SCADAPack E "RTU" I/O Device types access database points configured in the RTU.

Both physical I/O points and derived points are accessed using an "RTU" I/O device.

The I/O device configuration used in a IEC 61131-3 Resource need not necessarily correspond to the RTU physical I/O card arrangements.

**SCADAPack E "RTU" I/O Devices**

| Device Name | IEC 61131-3 Data Type | Max. no. of Channels | RTU Points supported |
|---|---|---|---|
| RTU_BIN_READ | BOOL *VarInput* variables | 100 | Physical Input, Derived & System Binary Objects. |
| RTU_BIN_WRITE | BOOL *VarOutput* variables | 100 | Physical Output, Derived & System Binary Objects |
| RTU_BIN_READ_OUTPUT | BOOL *VarInput* variables | 100 | Physical Output, Derived & System Binary Objects with Status |
| RTU_RAW_READ | DINT *VarInput* variables | 100 | Physical Input &, Derived Integer Objects. |
| RTU_ENG_READ | REAL *VarInput* variables | 100 | Physical Input & Derived Floating Point Objects |
| RTU_RAW_WRITE | DINT *VarOutput* variables | 100 | Physical Output &, Derived Integer Objects |
| RTU_ENG_WRITE | REAL *VarOutput* variables | 100 | Physical Output &, Derived Floating Point Objects |
| RTU_RAW_READ_OUTPUT | DINT *VarInput* variables | 100 | Physical Output &, Derived Integer Object with Status |
| RTU_ENG_READ_OUTPUT | REAL *VarInput* variables | 100 | Physical Output &, Derived Floating Point Object with Status |

| | | | |
|---|---|---|---|
| RTU_COUNTER_READ | UDINT *VarInput* variables | 100 | Counter Input Objects |
| RTU_STRING_WRITE | STRING *VarOutput* variables | 1 | System String points |

### 8.3.1 Binary (Digital) RTU I/O Devices

## I/O Devices for Binary (Digital) Input Points

Physical & Derived RTU binary points may be read through **RTU_BIN_READ** I/O Devices. Where an IEC 61131-3 Resource attaches a BOOL (*VarInput* Direction) variable to a RTU_BIN_READ I/O device, the *Current State Property* of the binary point will be read into the IEC 61131-3 Resource variable. If the binary point is a Physical Binary I/O address, the physical binary input channel corresponding to that address is read.

Binary point database values, including those read into an IEC 61131-3 resource through an I/O Device, can be preset from an IEC 61131-3 Resource by using the **SETPNTB** function.

## I/O Devices for Binary (Digital) Output Points

Physical RTU binary outputs have two sets of IEC 61131-3 interfaces. The state of a binary outputs are controlled through **RTU_BIN_WRITE** I/O devices. The feedback status of binary outputs are read through **RTU_BIN_READ_OUPUT** I/O devices.

Derived RTU binary points are controlled through RTU_BIN_WRITE I/O devices. The feedback status of derived binary points is read into RTU_BIN_READ I/O Devices.

Where an IEC 61131-3 Resource attaches a BOOL (*VarOutput* Direction) variable to a RTU_BIN_WRITE I/O device, the *Current State Property* of the binary point will be controlled from the IEC 61131-3 BOOL variable.

## Counter Input Devices

The *Current Integer Value* property of a physical counter input can be read through an **RTU_COUNTER_READ** I/O Device. The IEC 61131-3 Resource attaches a UDINT (Unsigned 32-bit Integer) variable to this I/O Device type.

Counter point database values can be preset from an IEC 61131-3 Resource by using the **SETPNTC** function.

### 8.3.2 Analog RTU I/O Devices

## I/O Devices for Analog Input Points

Physical and Derived RTU Analog points may be read through **RTU_RAW_READ** and **RTU_ENG_READ** I/O Devices. Where an IEC 61131-3 Resource attaches a DINT (*VarInput* Direction) variable to a RTU_RAW_READ I/O device, the *Current Integer Value property* of the analog point will be read into the IEC 61131-3 Resource variable. If the analog point is a Physical Analog I/O address, the physical analog input channel corresponding to that address is read.

Where an IEC 61131-3 Resource attaches a REAL (*VarInput* Direction) variable to a RTU_ENG_READ I/O device, the *Current Eng. Value property* of the analog point will be read into the IEC 61131-3 Resource variable

Both DINT and REAL IEC 61131-3 analog variables cannot be mixed on the same input I/O Device. Use separate I/O Devices for DINT and REAL variables, or use the ANY_TO_DINT or ANY_TO_REAL conversion functions if this is necessary.

IEC 61131-3 DINT variables contain signed 32-bit numbers. The value of a DINT *VarInput* variable will be the physical analog input variable in the range RAW-MIN to RAW-MAX as configured in the point's database attributes.

IEC 61131-3 REAL variables contain 32-bit floating point numbers. For a REAL *VarInput* variable, variables will be in the range ENG-MIN to ENG-MAX as configured in the point's database attributes.

Analog point database values, including those read into an IEC 61131-3 resource through an I/O Device, can be preset by using a **SETPNTSL** or **SETPNTF** function.

## I/O Devices for Analog Output Points

Physical RTU Analog Outputs have two sets of IEC 61131-3 interfaces. The value of physical analog outputs is controlled through **RTU_RAW_WRITE** and **RTU_ENG_WRITE** I/O devices. The feedback status of analog outputs are read through **RTU_RAW_READ_OUPUT** and **RTU_ENG_READ_OUTPUT** I/O devices.

Derived RTU Analog points are controlled through **RTU_RAW_WRITE** and **RTU_ENG_WRITE** I/O devices. The feedback status of derived analog points is read into **RTU_RAW_READ** and **RTU_ENG_READ** I/O Devices.

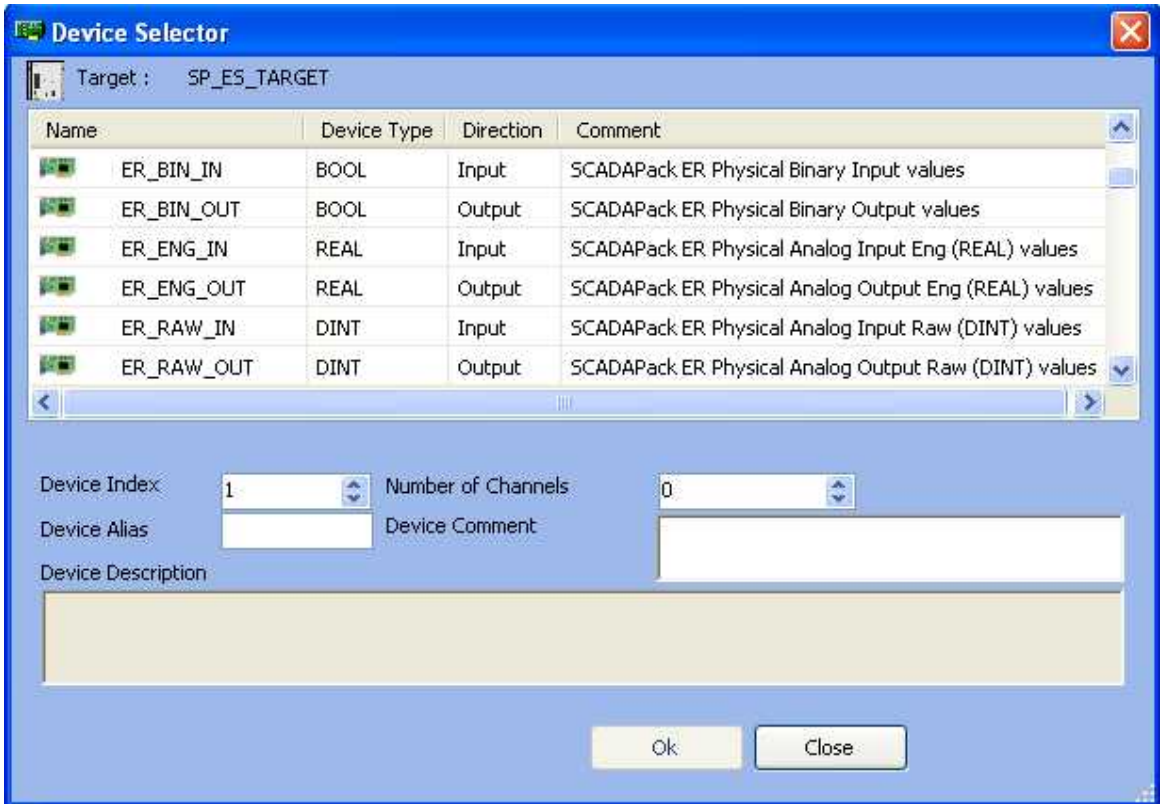
Where a IEC 61131-3 Resource connects a DINT *VarOutput* variable to an RTU_RAW_WRITE I/O device , the *Current Integer Value* property of the analog point will be controlled from the IEC 61131-3 variable. The analog point's *Current Integer Value* property, RAW-MIN, RAW-MAX, ENG-MIN & ENG-MAX scaling attributes will be used to automatically calculate the *Current Eng.Value* property of the point.

Where a IEC 61131-3 application attaches a REAL (floating point) *VarOutput* variable to an RTU_ENG_WRITE I/O device, the *Current Eng. Value* property of the analog point will be controlled from the IEC 61131-3 variable. The analog point's *Current Eng. Value* property RAW-MIN, RAW-MAX, ENG-MIN & ENG-MAX scaling attributes will be used to automatically calculate the *Current Integer Value* property of the analog point.

Both DINT and REAL IEC 61131-3 analog variables cannot be mixed on the same output I/O Device. Use separate I/O Devices for DINT and REAL variables, or use the ANY_TO_DINT or ANY_TO_REAL conversion functions if this is necessary.

### 8.3.3 SCADAPack ER I/O Devices

These I/O devices are only supported on SCADAPack ER RTUs. Resources referencing these I/O Devices will **not** start on RTUs other than SCADAPack ER RTU. These I/O Devices are only available in the Workbench I/O *Device Selector* window if a ***SCADAPack ES/ER Template*** 12 was chosen for the project.




*SCADAPack ER I/O Device Selector*

SCADAPack ER I/O devices **reference physical channels directly**, as opposed to referencing a specific I/O channel by point number. The SCADAPack ER I/O devices provided in the SCADAPack Workbench are listed in the following table.

**SCADAPack ER I/O Devices**

| Device Name | IEC 61131-3 Data Type | Max. no. of Channels |
|---|---|---|
| ER_BIN_IN | BOOL<br>*VarInput* variables | 32 |
| ER_BIN_OUT | BOOL<br>*VarOutput* variables | 16 |
| ER_ENG_IN | REAL<br>*VarInput* variables | 16 |
| ER_ENG_OUT | REAL | 4 |

| | | |
|---|---|---|
| | *VarOutput* variables | |
| ER_RAW_IN | DINT<br>*VarInput* variables | 16 |
| ER_RAW_OUT | DINT<br>*VarOutput* variables | 4 |

The SCADAPack ER I/O Devices reference the respective physical I/O cards by specifying a *Slot_Number* field. The *Slot_Number* field is set via user configuration through the I/O device parameters. These are set in the Workbench Resource's *I/O Device* window.

The required fields are described as follows:

***Slot_Number***: specifies the I/O card slot on the SCADAPack ER rack
1 = I/O Card Slot 1
2 = I/O Card Slot 2, etc.

The default value is 1 (i.e. I/O Card Slot 1).


*SCADAPack ER Device Parameters*

A valid I/O card configuration needs to be loaded into the SCADAPack ER RTU prior to loading an IEC 61131-3 application that references SCADAPack ER I/O devices, otherwise the I/O device will not be opened and the **IEC 61131-3 application WILL NOT START**. This is done using SCADAPack E Configurator by assigning an I/O card and writing the Configurator file changes onto the RTU. A RTU restart is required after these configuration details have been written to the RTU. See *SCADAPack E Configurator User Manual* for details.

#### 8.3.3.1 Digital I/O Devices

## Digital Output Device: ER_BIN_OUT

The ER_BIN_OUT I/O device references a physical relay output card by specifying a *Slot_Number* field (see ***SCADAPack ER I/O Devices*** 59 for detailed descriptions of these fields). The Channel number in the Workbench Resource's *I/O Device* window corresponds to the physical channel number on the SCADAPack ER I/O card.

Where a IEC 61131-3 Resource connects a BOOL *VarOut* variable to a ER_BIN_OUT output device, the state of the corresponding digital relay will be controlled from the IEC 61131-3 variable. If there is a physical digital output configuration point associated with this physical channel, the *Current State* of this configuration point will be updated after the successful control of the relay output.

Controls issued to SCADAPack ER relay output cards resulting from **attached** variables changing state, are issued as complete I/O card controls. This allows that any simultaneous state changes at the IEC 61131-3 I/O Device level, are executed simultaneously at the SCADAPack ER relay output card.

The ER_BIN_OUT output device can be opened only if there is valid I/O card configuration loaded into the SCADAPack ER RTU. Unlike the "RTU" output devices, it is NOT necessary that there are physical digital output configurations points associated with the physical channels referenced by the output device.

## Digital Input Device: ER_BIN_IN

The ER_BIN_IN I/O device references a physical binary input card by specifying a *Slot_Number* field (see Section ***SCADAPack ER I/O Devices*** 59 for detailed descriptions of these fields). The Channel number in the Workbench Resource's *I/O Device* window corresponds to the physical channel number on the SCADAPack ER I/O card.

Unlike the ER_BIN_OUT I/O device, there needs to be point database configuration points associated with the physical channels referenced by the ER_BIN_IN I/O device's *Slot_Number* fields for proper operation.

Where a IEC 61131-3 Resource connects a BOOL *VarInput* variable to a ER_BIN_IN I/O device, the *Current State Property* of the digital point will be read into the IEC 61131-3 variable. The ER_BIN_IN I/O device may be successfully opened if there is a valid I/O card configuration loaded into the SCADAPack ER , and there is at least 1 physical binary input configuration point associated with the given I/O card.

#### 8.3.3.2 Analog I/O Devices

## Analog Output Devices:

### ER_RAW_OUT

The ER_RAW_OUT I/O Device references a physical analog output card by specifying a *Slot_Number* field (the slot number that the card is installed on a SCADAPack ER RTU rack). The Channel number in the Workbench Resource's *I/O Device* window corresponds to the physical channel number on the SCADAPack ER I/O card.

Where an IEC 61131-3 Resource attaches an DINT *VarOutput* variable to an ER_RAW_OUT output device, the *Current Integer Value* property of the associated analog point will be controlled from the IEC 61131-3 variable. If there is a physical analog output configuration point associated with this physical channel, the *Current Integer Value* and *Engineering Value* of this configuration point will be updated after the successful control of the relay output.

The ER_RAW_OUT output device may be successfully opened if there is valid I/O card configuration loaded into the SCADAPack ER RTU. Unlike standard RTU point output devices (e.g. RTU_RAW_WRITE), it is NOT necessary that there are physical analog output configurations points associated with the physical channels referenced by the ER_RAW_OUT output device.

Both DINT and REAL IEC 61131-3 analog variables cannot be mixed on the same output I/O Device. Use the ANY_TO_DINT or ANY_TO_REAL conversion functions if this is required.

### ER_ENG_OUT

The ER_ENG_OUT I/O device references a physical analog output card by specifying a *Slot_Number* field (the slot number that the card is installed on a SCADAPack ER RTU rack). The Channel number in the Workbench Resource's *I/O Device* window corresponds to the physical channel number on the SCADAPack ER I/O card.

Where an IEC 61131-3 Resource attaches a REAL *VarOutput* variable to an ER_ENG_OUT output device, the *Current Eng Value* property of the associated analog point will be controlled from the IEC 61131-3 variable. If there is a physical analog output configuration point associated with this physical channel, the *Current Eng Value* and *Current Integer Value* of this configuration point will be updated after the successful control of the analog output.

The ER_ENG_OUT output device may be successfully opened if there is valid I/O card configuration loaded into the SCADAPack ER RTU. Unlike standard RTU point output devices (e.g. RTU_ENG_WRITE), it is NOT necessary that there are physical analog output configurations points associated with the physical channels referenced by the ER_ENG_OUT output device.

Both DINT and REAL IEC 61131-3 analog variables cannot be mixed on the same output I/O Device. Use the ANY_TO_DINT or ANY_TO_REAL conversion functions if this is required.

## Analog Input Devices:

### ER_RAW_IN

The ER_RAW_IN I/O device references a physical analog input card by specifying a *Slot_Number* field (the slot number that the card is installed on a SCADAPack ER RTU rack). The Channel number in the Workbench Resource's *I/O Device* window corresponds to the physical channel number on the SCADAPack ER I/O card.

Unlike the ER_RAW_OUT device, there needs to be point database configuration points associated with

the physical channels referenced by the ER_RAW_IN input device for proper operation.

Where an IEC 61131-3 Resource attaches a DINT *VarInput* analog variable to an ER_RAW_IN input device, the *Current Integer Value* property of the associated analog point will be read into the ISaGRAF variable.

The ER_RAW_IN input device may be successfully opened if there is a valid I/O card configuration loaded into the SCADAPack ER RTU, and there is at least 1 physical analog input configuration point associated with the given I/O card.

Both DINT and REAL IEC 61131-3 analog variables cannot be mixed on the same input I/O Device. Use the ANY_TO_DINT or ANY_TO_REAL conversion functions if this is required.

## ER_ENG_IN

The ER_ENG_IN I/O Device references a physical analog input card by specifying a *Slot_Number* field (the slot number that the card is installed on a SCADAPack ER RTU rack). The Channel number in the Workbench Resource's *I/O Device* window corresponds to the physical channel number on the SCADAPack ER I/O card.

Unlike the ER_ENG_OUT output device, there needs to be point database configuration points associated with the physical channels referenced by the ER_ENG_IN I/O Device for proper operation.

Where an IEC 61131-3 Resource attaches a REAL (floating point) *VarInput* analog variable to an ER_ENG_IN input device, the *Current Eng Value* property of the associated analog point will be read into the ISaGRAF variable.

The ER_ENG_IN input device may be successfully opened if there is a valid I/O card configuration loaded into the SCADAPack ER RTU, and there is at least 1 physical analog input configuration point associated with the given I/O card

Both DINT and REAL IEC 61131-3 analog variables cannot be mixed on the same input I/O Device. Use the ANY_TO_DINT or ANY_TO_REAL conversion functions if this is required.

## 8.4 Remote Control Interlock & Hold On Stop

### Remote Control Interlock

By default, RTU output points (Binary & Analog, Physical & Derived) attached to an Output I/O device are under ISaGRAF control while an IEC 61131-3 Resource is executing. Attempts to control an output point that is under *ISaGRAF control* will normally be unsuccessful. However, a “Remote Interlock” point may be associated with each output point.

If a "Remote Interlock" point is defined and configured for an output point, but is inactive, the IEC 61131-3 resource retains control of the output point. Communication requests to the output point (e.g. from DNP3, IEC60870-5 protocols, Modbus, etc) are rejected while the application is running.

If a “Remote Interlock” point is defined and configured for an output point, but is active, then the IEC 61131-3 resource does not have control of the physical output point. Instead, communication requests to physical outputs are accepted to the point.

### Hold On Stop

The SCADAPack Workbench Output I/O Devices support a **hold_on_stop** parameter.

This BOOL parameter allows the user to identify how outputs are processed on a shutdown (or restart) of the IEC 61131-3 application.

This applies for local RTU physical output points on the SCADAPack ES/ER RTUs.

The hold on stop parameter is **not used** on the SCADAPack 300E RTUs even if it is available on a Workbench I/O device.

See ***Stopping an IEC 61131-3 Resource*** 52 for a description of the effect of stopping an IEC 61131-3 application.

This Hold On Stop Output I/O device parameter defaults to *false* (meaning outputs will be reset off or reset to 0 when the application stops).

When set to true, the local physical outputs of SCADAPack ES/ER RTUs are held at their current values until the database point is updated to some other value (e.g. by a new application, DNP3 / IEC60870-5-101 / Modbus control, etc).

## 8.5 PLC Device I/O Devices

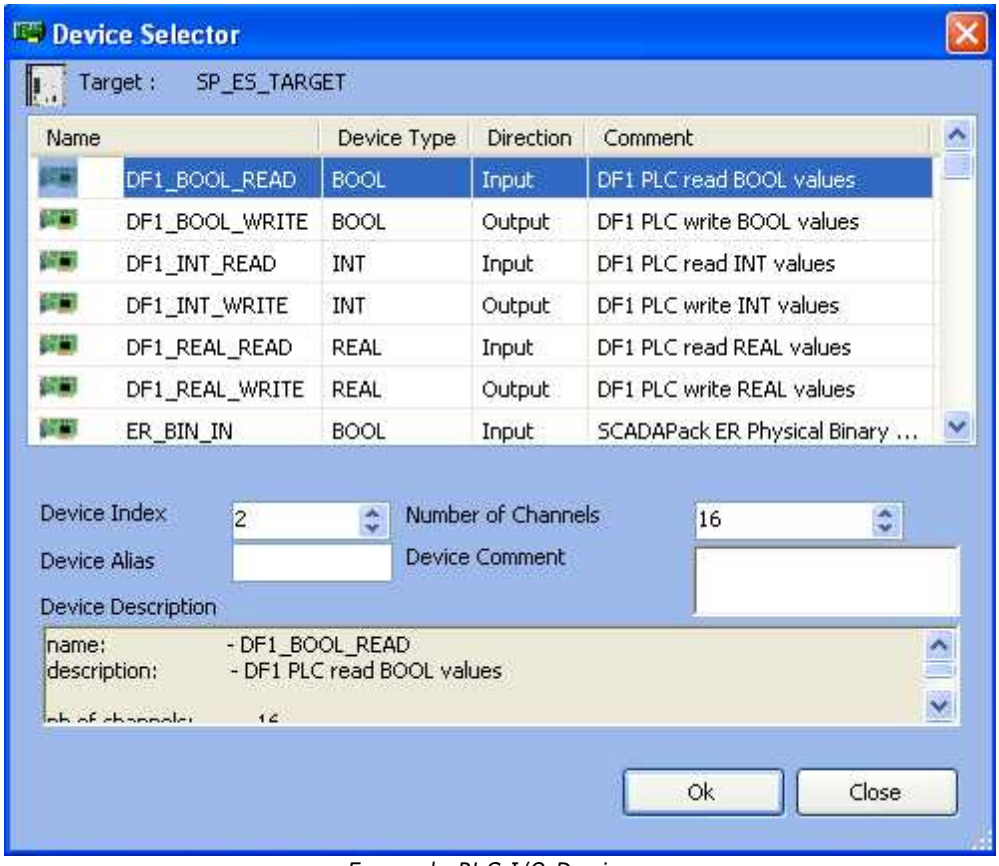
As an extension of the data interface provided by the SCADAPack E RTU, access by IEC 61131-3 Resources to external PLC or peripheral device data is supported.

Standard "RTU" I/O devices can access RTU I/O and the point database, while PLC I/O Devices allow data to be extracted from external devices such as PLCs,

External peripheral data (e.g. from PLCs) is cached internally by the RTU to maximize IEC 61131-3 application performance. Access to this cached device data is restricted to IEC 61131-3 and is termed PLC Device data. Direct access to PLC device data through DNP3 / IEC60870 or other protocols requires additional application logic to copy the peripheral data to/from the RTU point database.

The PLC device that can be accessed via the PLC I/O Device depends on the PLC or peripheral device drivers supported by the RTU Operating System firmware.

LED(s) on the SCADAPack E RTU may indicate communication activity with external peripheral device (s). For more information see the relevant RTU *Hardware User Manual*.




*Example PLC I/O Devices*

**TIP:** When an IEC 61131-3 application Resource starts, particularly if there is a large number of PLC Device I/O devices, or if a PLC is not responding, the Workbench debugger may not connect to the SCADAPack E target immediately. In this case please wait a little while for the application to work through the PLC device updates before attempting to re-establish a

debugger connection.

Different PLC Device I/O Devices are provided for different types of PLC data.

For example: read PLC value registers (analog input device), write PLC coils (boolean output device), read PLC accumulated data (analog input device). The different types of I/O devices available and ranges of PLC data that can be accessed depend on the individual PLC driver. The following section details a summary of the SCADAPack E RTU's MODBUS PLC driver. For detailed information on Modbus and other drivers, see the relevant *SCADAPack E* PLC Device *Interface* manual.

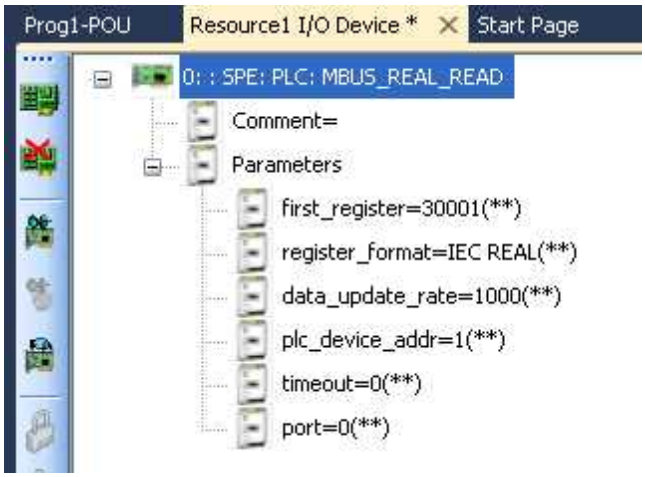
ISaGRAF PLC Device I/O devices access data in the following way:

- a READ PLC I/O Device normally corresponds to a read access to PLC data
- a WRITE PLC I/O Device corresponds to a write-only access to PLC data
- Serial communication with external devices, such as PLCs, is made through the SCADAPack E RTU port(s) configured as *PLC Device*.
- IEC 61131-3 function blocks: **MBUSCTRL, MTCPCTRL & DF1CTRL** can be used to control read & write communications to the PLC device whose communication is defined by I/O devices. Disabling read operations affects PLC device Input data. Disabling write operations affects PLC device Output data.
- Up to a total of 200 PLC I/O Devices can be defined in total for *PLC Device* communication ports and IEC 61131-3 Resources. Multiple *PLC Device* serial ports, as well as TCP/IP channels for some PLC devices, can be used for PLC peripheral communication.

### 8.5.1 Reading PLC Registers

PLC I/O input devices typically require user configuration through the I/O device parameters. These are set as part of the Workbench application and are entered into the I/O device parameter fields within the Workbench Resource's *I/O Device* Window.

The SCADAPack E RTU can communicate with Allen Bradley DF1 PLC's, Modbus PLC peripheral devices via the serial 'MODBUS RTU', Open Modbus/TCP or Modbus RTU in TCP protocols.




*Example PLC I/O Device Parameters*

Typical PLC I/O Device parameter fields are as follows:

***first_register**:* specifies the PLC Device data registers to access when reading PLC data into ISaGRAF variables. The PLC data type accessed is specific to the PLC Device I/O device and address. This value is usually the PLC's data (or register) address.

***register_format**:* specifies the PLC data register type. The following data types are supported: IEC DISCRETE, 984 DISCRETE, IEC UINT IEC INT, IEC DINT, IEC REAL, SWAP REAL. See specific PLC driver interface manuals for more information.

***data_update_rate**:* The units for this parameter vary depending on the type of PLC device. For example this may be a setting in milliseconds for a directly connected device, or in minutes for a low power type device (see the *SCADAPack E Target 5 Modbus Communication Interfaces* manual). As the SCADAPack E RTU needs to extract the data for the I/O device from the PLC or peripheral device, this sets the rate at which the data is extracted. Individual I/O devices may have different data update rates allowing prioritization of data extracted from a PLC device. The RTU may not be able to read requested PLC data within the time set by the data update rate depending on the quantity of data to be read, rate of write requests and PLC communication speed. In this case the update rates will be slower.

***plc_device_addr**:* This parameter specified the PLC device address. Some PLC device drivers support multi-drop PLC devices on the same communication channel, or have unique addressing identifiers. Where the RTU driver provides multi-drop support, an IEC 61131-3 application may access data from any of the locally multi-dropped devices. A separate I/O device will be required for each device.

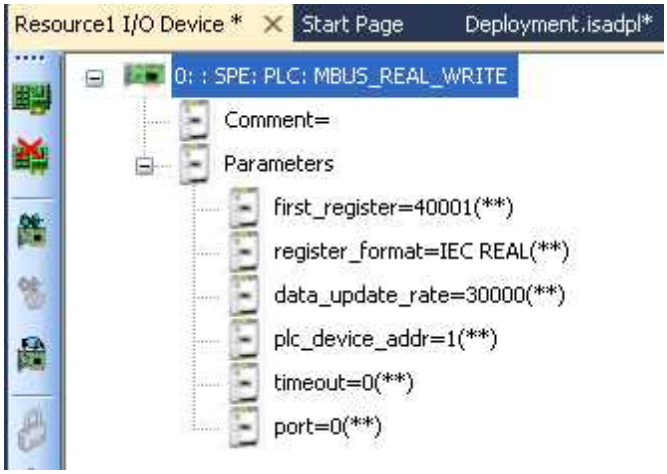
***timeout**:* PLC device drivers with comprehensive I/O device interfaces may provide a parameter for specifying the communications timeout on an individual I/O device (i.e. the timeout applies to communications associated with that device). Where this value is "0", the PLC device driver will use a default timeout of 1200 milliseconds. The units for this field are dependent upon the PLC device driver. Units may be, for example, milliseconds, seconds, minutes, etc.

***Port**:* This parameter is only available on certain PLC I/O device drivers. Where present, it defines which of multiple RTU *PLC Device* ports will be used to communicate with the PLC or peripheral device. PLC I/O Devices not including this parameter can only be used when a single *PLC Device* port is configured on the SCADAPack E RTU.

### 8.5.2 Writing PLC Registers

PLC I/O output devices require user configuration through the I/O device parameters. These are set as part of the Workbench application and are entered into the I/O device parameter fields within the Workbench Resource's *I/O Device* Window.

The SCADAPack E RTU can communicate with Allen Bradley DF1 PLC's, Modbus PLC peripheral devices via the serial 'MODBUS RTU', Open Modbus/TCP or Modbus RTU in TCP protocols.




*Example PLC I/O Device Parameters*

Typical PLC I/O Device parameter fields are as follows:

***first_register**:* specifies the PLC Device data registers to access when writing from ISaGRAF variables to PLC data. The PLC data type accessed is specific to the PLC Device I/O device and address. This value is usually the PLC's data (or register) address.

***register_format**:* specifies the PLC data register type. Currently *IEC UINT, INT, DINT, IEC REAL & SWAP REAL* types are supported for analog devices and *IEC DISCRETE* type is supported for Boolean devices. See specific PLC driver interface manuals for more information.

***plc_device_addr**:* This parameter specifies the PLC device address. Some PLC device drivers support multi-drop PLC devices on the same communication channel, or have unique addressing identifiers. Where the RTU driver provides multi-drop support, an IEC 61131-3 application may access data from any of the locally multi-dropped devices. A separate I/O device will be required for each device.

***data_update_rate**:* The unit for this parameter is driver specific, and configures the rate at which the data for the I/O Device is written to the PLC. Between "data update rate" periods, data is written to the PLC only when the IEC 61131-3 output variable values change. Individual I/O devices may have different must write rates allowing prioritization of data sent to a PLC Device.

***timeout**:* PLC device drivers with comprehensive I/O device interfaces may provide a parameter for specifying the communications timeout on an individual I/O device (i.e. the timeout applies to communications associated with that device). Where this value is "0", the PLC device driver will use a default timeout. The units for this field are dependent upon the PLC device driver. Units may be, for

example, milliseconds, seconds, minutes, etc.

***port***: This parameter is only available on certain PLC I/O device drivers. Where present, it defines which of multiple RTU *PLC Device* ports will be used to communicate with the PLC or peripheral device. If only one *PLC Device* port is configured, this field is ignored. PLC I/O Devices not including this parameter can only be used when a single *PLC Device* port is configured on the SCADAPack E RTU.

### 8.5.3 I/O Device Status

The SCADAPack E RTU checks for data being written to the PLC by an IEC 61131-3 I/O device driver, before the PLC I/O Input Device data is retrieved.

Communication requests made by the SCADAPack E RTU to the PLC are asynchronous to the scanning of the IEC 61131-3 Resource, but data within the resource remains consistent duration of a scan cycle.

To assist with debugging of PLC I/O Device communication, the SCADAPack E RTU provides two types of *analog* system points which provide useful information:

1. PLC Communication Status

Status available for the first 60 PLC I/O Devices used by IEC 61131-3 *Resource Number* 1 and the first 14 PLC I/O Devices used by *Resource Number* 2.

2. Cache Time (seconds)

- for PLC I/O Input Devices, these system points represents the age of the data since the last update
- these system points are not updated for PLC I/O Output Devices (0).

SCADAPack E RTU analog system points 53300 to 53419 are set aside for the PLC Device communication status information for up to 60 PLC I/O Devices in IEC 61131-3 *Resource Number* 1. Points 53422 to 53449 are set aside for the PLC Device communication status information for up to 14 PLC I/O Devices in *Resource Number* 2. A pair of consecutive points represent the PLC Communication status ('COMMS_STATUS') and the age of the data ('UPDATE_TIME') for each PLC I/O Device (refer to table below). Non 'PLC' I/O devices (i.e. "RTU" devices), are **not** included in the consecutive count.

This data is accessible externally to the RTU (via a communications protocol accessing these points) or from an IEC 61131-3 resource, using RTU_RAW_READ I/O devices. These points are only accessible if PLC Device functionality is enabled.

**'Comms Status' & 'Cache Time' System Point Numbers for IEC 61131-3 *Resource Number* 1**

| Consecutive PLC I/O Device | Comms_Status Analog System Point No. | Cache_Time Analog System Point No. |
|---|---|---|
| 1 | 53300 | 53301 |
| 2 | 53302 | 53303 |
| 3 | 53304 | 53305 |
| 4 | 53306 | 53307 |
| 5 | 53308 | 53309 |
| 6 | 53310 | 53311 |
| 7 | 53312 | 53313 |
| 8 | 53314 | 53315 |
| 9 | 53316 | 53317 |
| 10 | 53318 | 53319 |

| Consecutive PLC I/O Device | Comms_Status Analog System Point No. | Cache_Time Analog System Point No. |
|---|---|---|
| 11 | 53320 | 53321 |
| 12 | 53322 | 53323 |
| 13 | 53324 | 53325 |
| 14 | 53326 | 53327 |
| 15 | 53328 | 53329 |
| 16 | 53330 | 53331 |
| 17 | 53332 | 53333 |
| 18 | 53334 | 53335 |
| 19 | 53336 | 53337 |
| 20 | 53338 | 53339 |
| 21 | 53340 | 53341 |
| 22 | 53342 | 53343 |
| 23 | 53344 | 53345 |
| 24 | 53346 | 53347 |
| 25 | 53348 | 53349 |
| 26 | 53350 | 53351 |
| 27 | 53352 | 53353 |
| 28 | 53354 | 53355 |
| 29 | 53356 | 53357 |
| 30 | 53358 | 53359 |
| 31 | 53360 | 53361 |
| 32 | 53362 | 53363 |
| 33 | 53364 | 53365 |
| 34 | 53366 | 53367 |
| 35 | 53368 | 53369 |
| 36 | 53370 | 53371 |
| 37 | 53372 | 53373 |
| 38 | 53374 | 53375 |
| 39 | 53376 | 53377 |

| Consecutive PLC I/O Device | Comms_Status Analog System Point No. | Cache_Time Analog System Point No. |
|---|---|---|
| 40 | 53378 | 53379 |
| 41 | 53380 | 53381 |
| 42 | 53382 | 53383 |
| 43 | 53384 | 53385 |
| 44 | 53386 | 53387 |
| 45 | 53388 | 53389 |
| 46 | 53390 | 53391 |
| 47 | 53392 | 53393 |
| 48 | 53394 | 53395 |
| 49 | 53396 | 53397 |
| 50 | 53398 | 53399 |
| 51 | 53400 | 53401 |
| 52 | 53402 | 53403 |
| 53 | 53404 | 53405 |
| 54 | 53406 | 53407 |
| 55 | 53408 | 53409 |
| 56 | 53410 | 53411 |
| 57 | 53412 | 53413 |
| 58 | 53414 | 53415 |
| 59 | 53416 | 53417 |
| 60 | 53418 | 53419 |

**'Comms Status' & 'Cache Time' System Point Numbers for IEC 61131-3 *Resource Number 2***

| Consecutive PLC I/O Device | Comms_Status Analog System Point No. | Cache_Time Analog System Point No. |
|---|---|---|
| 1 | 53422 | 53423 |
| 2 | 53424 | 53425 |

| | | |
|---|---|---|
| 3 | 53426 | 53427 |
| 4 | 53428 | 53429 |
| 5 | 53430 | 53431 |
| 6 | 53432 | 53433 |
| 7 | 53434 | 53435 |
| 8 | 53436 | 53437 |
| 9 | 53438 | 53439 |
| 10 | 53440 | 53441 |
| 11 | 53442 | 53443 |
| 12 | 53444 | 53445 |
| 13 | 53446 | 53447 |
| 14 | 53448 | 53449 |

These system points are only accessible if PLC Device functionality is enabled through a *PLC Device* port setting or *Modbus/IP (Client)* setting.

**PLC Comms Status values**

**0** = Normal
**101** = Unknown device
**102** = Illegal Data Count
**103** = Illegal Data Address
**104** = Device Timeout
**105** = Read/Write lock unsuccessful
**106** = Invalid message
**107** = Device Busy
**108** = Data value out of range

PLC Input I/O Device data is updated by the SCADAPack E RTU when it communicates with the PLC device. The PLC communication status is updated if there is a status code returned from the PLC or no response from the PLC after a data request by the RTU. Status codes are presented in the ***Table*** 73 above. The value in each status register is cleared by the RTU upon successful communication sessions. Variables within an IEC 61131-3 resource can be used to log transient status codes.

PLC Output I/O Device data is written to the PLC when an IEC 61131-3 resource changes the value of a variable attached to the I/O device Channel. In addition, output data is written to the PLC under the following conditions:

- When the IEC 61131-3 resource starts, output data is written
- If the PLC does not respond to a control, it is resent until it is responded
- I/O devices with a *data_update_rate* parameter output data at this rate

- Output data is rewritten at a background update rate

SCADAPack E RTU Analog System point 53420 controls the background update rate of PLC Output I/O Devices on the RTU. Its default value is 60 seconds. It may be adjusted by the user dynamically or specified in an RTU configuration, and is a non-volatile RTU system point. The background updates are disabled by setting the system point 53420 value to 0 (zero). This may be used to optimize the PLC Device communications bandwidth where background writes are not appropriate or not necessary.

There are individual PLC I/O Devices available in the SCADAPack Workbench for different types of PLC types and different types of data within the same PLC device. Not every PLC data type for a particular PLC device may be accessible from the PLC I/O Devices. For more information see SCADAPack E Target 5 Modbus Communication Interfaces or SCADAPack E Target 5 DF1 PLC Interface manuals.

## 8.6 RTU Data via Function Blocks

SCADAPack E provides a mechanism separate from IEC 61131-3 I/O Devices (described in preceding sections) for accessing RTU data.

In general the function blocks require more processing capacity in the RTU compared with I/O Devices, but provide greater programming flexibility and access to more detailed RTU data.

Function blocks provide access to reading and writing current data values to/from the RTU point database, as well as access to point attributes not available via I/O Devices.

- The **GETPNTxx** functions and function blocks allow an IEC 61131-3 resource to read point current value data from the database (applies to Physical I/O, Derived data and System Points)
- The **SETPNTxx** functions allow an IEC 61131-3 resource to write point current value data to the database (applies to Physical Outputs, Derived data and System Points)
- The **RTUCROB** function blocks allow an IEC 61131-3 resource to have accurate pulse control of binary points (applies to Physical Binary Outputs and Derived Binary points)
- The **RDFLD_x** function blocks allow an IEC 61131-3 resource to read attribute and property fields from points in the RTU database
- The **SETATR_x** function blocks allow an IEC 61131-3 resource to set attributes of points in the RTU database
- The **RDREC_x** function blocks provide IEC 61131-3 resources with a set of commonly used attribute and property fields for points in the RTU database

IEC 61131-3 Arrays can be useful when using the above Functions and Function Blocks. For more information see ***Array Variables*** 34.

Details of these, and other SCADAPack Workbench functions blocks are described in the *SCADAPack E Target 5 Function Block Reference* manual.

## 8.7 I/O Device Pre-Processor

SCADAPack Workbench generates tag names (defined words) for variables associated with points in the RTU database. The names are used to access the point database through function blocks, rather than using the raw point number. When a variable is moved to a different point number, the value automatically changes and the IEC 61131-3 code using the name continues to access the correct point in the database.

### 8.7.1 Using the Pre-Processor

Building the solution, or a part of it, generates defined words from directly represented variable variables that are wired to RTU point database I/O devices. See I/O Devices Checked by the Pre-Processor 77 for a list of I/O devices used to generate defined words.

Defined words generated by previous builds are deleted first. The build deletes defined words that begin with `POINT_` including manually defined words.

Should the variable name break any of the rules outlined below, the Output window and Error List window will report the reason. The build will be unsuccessful. Correct the variable names and build the solution again.

A defined word has three parts: Name, Equivalent and Comment.

#### *Name*

The name identifies the resource, program, and variable name of the variable. In general the name takes the form

```
POINT_VariableName_ProgramName_ResourceName
```

These rules apply

- The total name must not exceed 128 characters.
- The name must contain only letter, digits and underscores
- The name must not end with an underscore
- The name must be unique
- The name must not contain a double underscore

If the variable name begins with an underscore character, the leading underscore of the variable is not included in the name generated by the pre-processor. This results in just one underscore between the `POINT` and `VariableName` parts of the name.

## Equivalent (Value)

The Equivalent (value) is the variable's point number. The point number is determined from I/O device's `First_Point_Number` parameter and the index of the channel. For example, if the `First_Point_Number` is 100 and the index of the channel is 2, the Equivalent (value) is 102.

## Comment

The comment is "Automatically generated by the RTU Point Pre-Processor"

## Examples

Some pre-processor naming examples are below.

### Example 1

With these variables

- Resource1 contains global variable `varRes1`
- Resource2 contains global variable `varRes2`

These defined words are created

- `POINT_varRes1_Resource1`
- `POINT_varRes2_Resource2`

### Example 2

With these variables

- Resource1 contains global variable `var1`
- Resource2 contains global variable `var1`

These defined words are created

- `POINT_var1_Resource1`
- `POINT_var1_Resource2`

### Example 3

With these variables

- Resource1 contains Program1 with `var1`
- Resource1 contains Program2 with `_var1`

These defined words are created

- `POINT_var1_Program1_Resource1`
- `POINT_var1_Program2_Resource1` (this example shows the removal of a leading underscore)

### Example 4

With these variables

- Resource1 contains SFC Program1 with child SFC program Program1A that contains `var1`
- Resource1 contains SFC Program1 with child SFC program Program1B that contains `var1`

These defined words are created

- `POINT_var1_Program1A_Program1_Resource1`
- `POINT_var1_Program1B_Program1_Resource1`

#### 8.7.2 I/O Devices Checked by the Pre-Processor

The pre-processor generates Defined Words for variables wired to these I/O devices.

- RTU_BIN_READ
- RTU_BIN_READ_OUTPUT
- RTU_BIN_WRITE
- RTU_COUNTER_READ
- RTU_ENG_READ
- RTU_ENG_READ_OUTPUT
- RTU_ENG_WRITE
- RTU_RAW_READ
- RTU_RAW_READ_OUTPUT
- RTU_RAW_WRITE
- RTU_STRING_WRITE
- RTU_BIN_WRITE_INPUT
- RTU_RAW_WRITE_INPUT
- RTU_ENG_WRITE_INPUT
- RTU_COUNTER_WRITE_INPUT

### 8.7.3 Function Blocks

Defined Words created by the pre-processor are useful with these function blocks.

- RDFLD_I
- RDFLD_R
- RDREC
- RDREC_AN
- RDREC_CN
- RDREC_DG
- RDSTRING
- SETATR_I
- SETATR_R

Defined Words created by the pre-processor can be used to access point data with these functions.

- GETPNTB
- GETPNTC
- GETPNTF
- GETPNTSL
- GETPNTSS
- GETPNTUS
- SETPNTB
- SETPNTF
- SETPNTSL
- SETPNTSS
- SETPNTUS

### 8.7.4 Differences with ISaGRAF 3 Workbench

A few differences with the pre-processor exist between ISaGRAF 3 Workbench and SCADAPack Workbench.

## Defined Word Name Length

The length of defined words has increased from 16 to 128 characters. This allows for a more useful prefix - `POINT_` - and includes the program and resource names to ensure uniqueness without having to perform various truncating and character replacement tricks.

## Prefix Configuration

Since the defined word name length is increased to 128 characters, there is no need to have a configurable prefix value. The prefix value used by the pre-processor is hard-coded to `POINT_`. There is no option to configure the pre-processor to use a suffix rather than a prefix.

## Array Variables

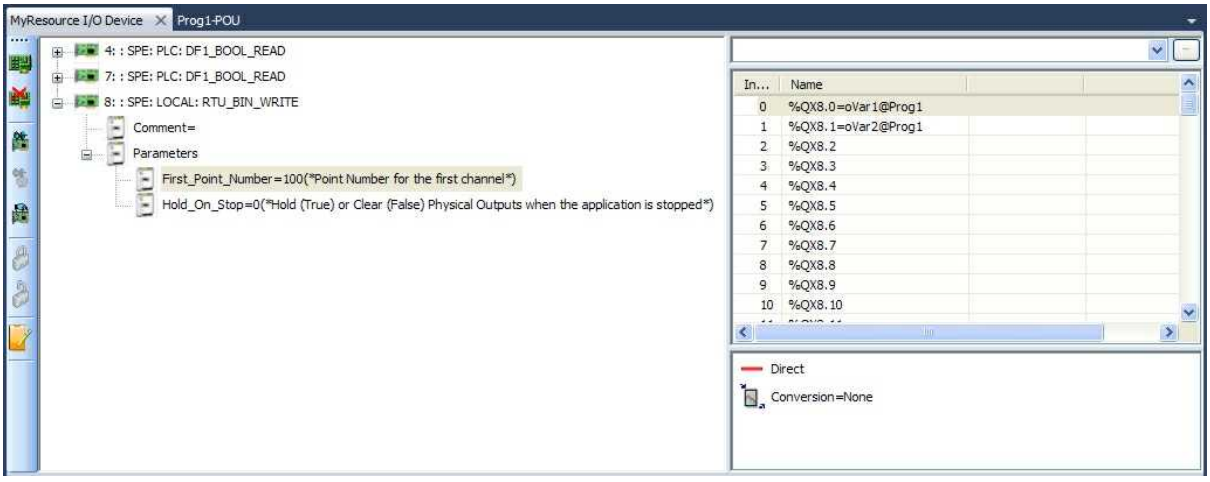
Array variables cannot be wired to I/O devices. The pre-processor does not generate Defined Words for array variables.

## Autocorrect of Variable Names

The pre-processor will not automatically truncate, replace characters or append digits to the end of defined word names as it did in ISaGRAF 3 Workbench.

### 8.7.5 Pre-Processor Example

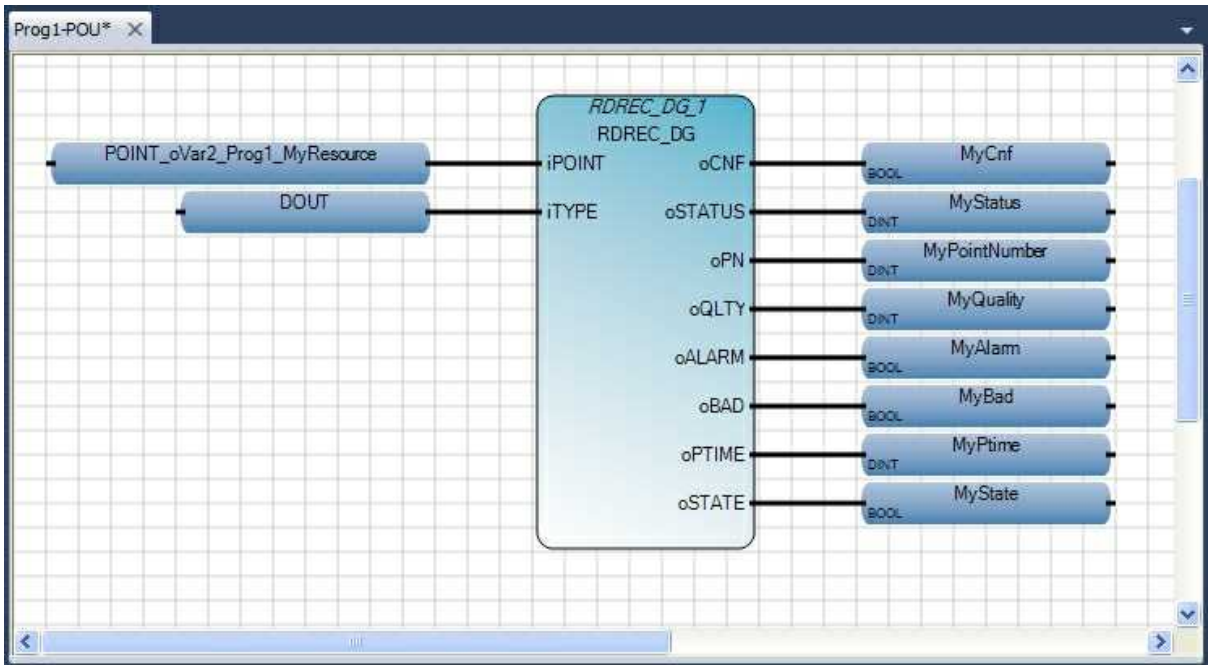
In this example, two variables are wired to the RTU_BIN_WRITE I/O device at device index 8. This I/O device is part of a resource named `MyResource`.




The Pre-Processor generates these Defined Words linking the DNP point to the I/O device.

- `POINT_oVar1_Prog1_MyResource` = 100
- `POINT_oVar2_Prog1_MyResource` = 101

The defined words can be used in a program as follows.




# 9 Remote Target Access

The following section detail how IEC 61131-3 applications and debugging on an SCADAPack E Smart RTU can be managed remotely from the SCADAPack Workbench.

## 9.1 Target Serial Communications

A SCADAPack Workbench debugger operates with the SCADAPack E RTU's on an RTU serial port configured as *ISaGRAF*. Operation is supported for RS232, RS422 and RS485.

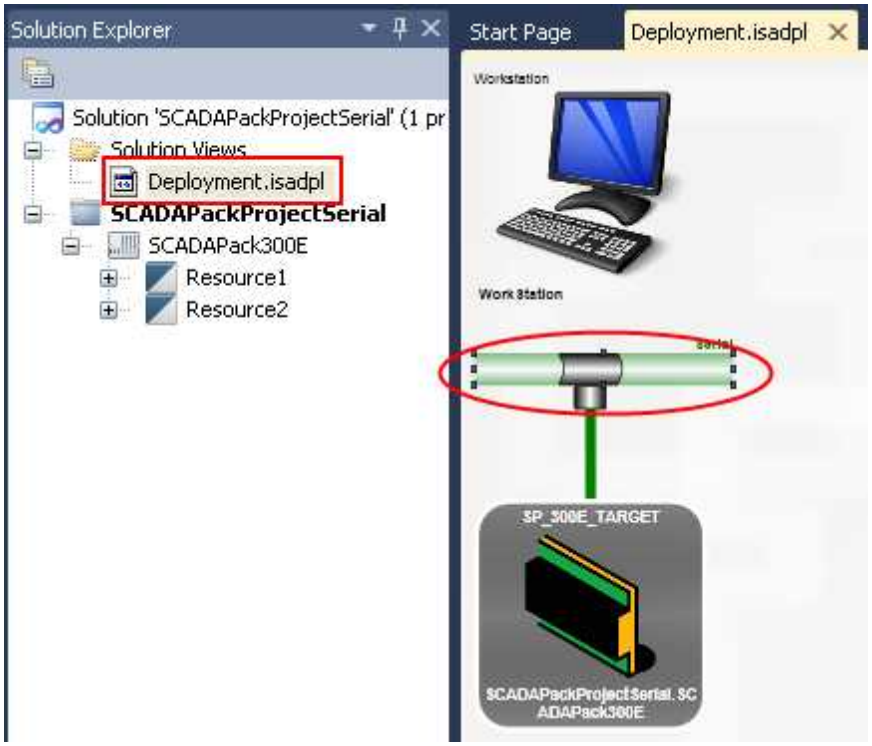
The type of connection used by the Workbench is a function of the project configuration. The SCADAPack Workbench software offers pre-configured ***Project Templates*** 121 for either Serial or TCP/IP connections.

> **TIP:** A maximum of one RTU serial port may be configured as *ISaGRAF* for local connection to a SCADAPack Workbench debugger. If a second Workbench connection is desired, use ISaGRAF via Ethernet (ETCP).

An IEC 61131-3 resource previously loaded into the RTU (via Workbench or File Transfer) will execute on the RTU regardless of whether an *ISaGRAF* port is configured.

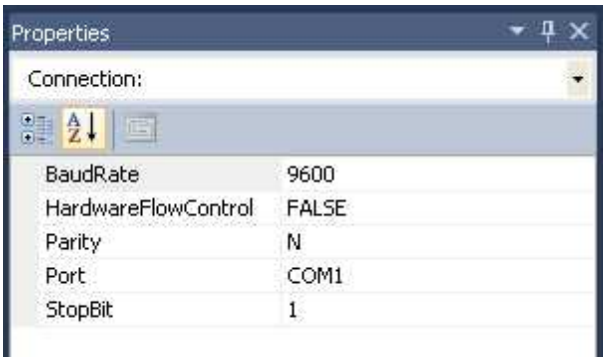
The SCADAPack Workbench project requires local configuration in order to communicate with a SCADAPack E RTU over a serial line.

Select the Workbench *Deployment View* window from the *View -> Deployment View* menu, or from the *Solution Explorer* window as shown below.




*Deployment View Window*

From the *Deployment View* window, select the *Network* connection (shown above as circled). The *Properties* window should then show the available Workbench serial port settings (see below).




The available serial port configuration fields are described below. These settings should match the *ISaGRAF* serial port configuration on the SCADAPack E RTU target.

| | |
|---|---|
| Port | The Workbench local communication port. The default value is COM1. |
| Baud Rate | The baud data transfer rate. The default value is 9600. |
| Parity | The type of parity used. Possible values are N for none, E for even, and O for odd;<br>The default value is N. |
| Stop Bit | The number of stop bits used to indicate the end of a transmission. Possible values are 1 or 2;<br>The default value is 1. |
| HardwareFlowControl | The control of the flow of data transmission between the network hardware. Possible values are True or False;<br>* Use the default value of False for SCADAPack E Smart RTU. |

## Adjusting the Serial Communication Timeout

Workbench will time out and abort communication if a response is not received from the SCADAPack E Smart RTU. In some applications it is necessary to use a longer serial communication timeout than the default value. The serial communication timeout cannot be adjusted from the Properties dialog. It can be adjusted by editing a text (XML) file. There are two parameters: a timeout and a multiplier.

To adjust the serial communication timeout:

- Locate the settings file
    - On Windows XP the file is C:\Documents and Settings\All Users\Application Data\Schneider Electric\6.1\Schneider Electric Gateway\OpcConfig.xml
    - On Windows Vista and Windows 7 the file is C:\ProgramData\Schneider Electric\6.1 \Schneider Electric Gateway\OpcConfig.xml

- Make a copy of the settings file (this can be used to restore the settings a mistake is made)
- Open the settings file with a text editor or XML editor
- This parameter adjusts the timeout. Edit the text highlighted below.

```
<!--Time-out in milliseconds to declare connection failure-->
<ConnectNoResponseTimeOut>10000</ConnectNoResponseTimeOut>
```

- This parameter adjusts the multiplier. Edit the text highlighted below.

```
<!--IXL Tx (10 sec.), Rx (4 sec.) time-out multiplication factor-->
<IxlTimeoutMultiFactor>3</IxlTimeoutMultiFactor>
```

- The timeout for receiving and connecting is equal to

```
IxlTimeoutMultiFactor * ConnectNoResponseTimeOut
```

## 9.2 TCP/IP Communications

SCADAPack Workbench can connect to an SCADAPack E RTU via Ethernet communications on a TCP/IP port. The Enhanced TCP/IP protocol, *ETCP* is the Workbench network driver used for communication with an IEC 61131-3 target on Ethernet.

The type of connection used by the Workbench is a function of the project configuration. The SCADAPack Workbench software offers pre-configured ***Project Templates*** 12 for either Serial or TCP/IP connections.

Either the RTU Ethernet or a serial PPP interface may be used for SCADAPack Workbench communications using TCP/IP.

To establish a TCP/IP connection between the Workbench and the RTU, the **ISaGRAF/TCP** service on the RTU needs to be enabled through the *TCP/IP* page on SCADAPack E Configurator software. Refer to the *SCADAPack E Configurator User Manual* for details.

The SCADAPack Workbench Network parameters need to be configured for a TCP/IP connection to a target RTU. Setting up a TCP/IP connection within the Workbench is described below.

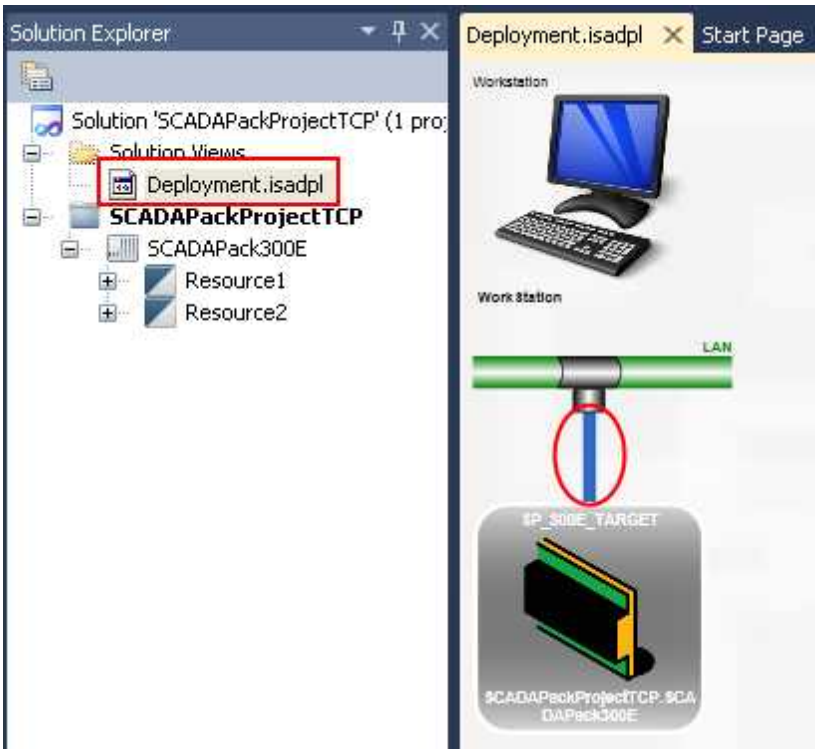
An IEC 61131-3 resource previously loaded into the RTU (via Workbench or File Transfer) will execute on the RTU regardless of whether a **ISaGRAF/TCP** service on the RTU is configured.


- ***Workbench Ethernet Settings*** 84
- ***TCP/IP Communications Server*** 85

### 9.2.1 Workbench Ethernet Settings

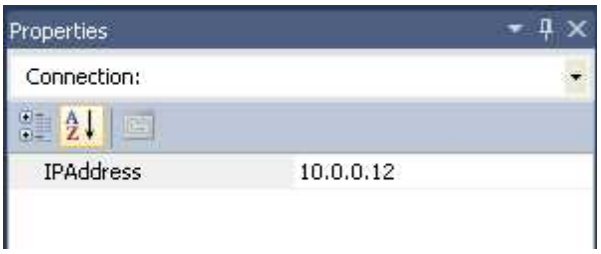
A SCADAPack Workbench project requires local configuration in order to communicate with a SCADAPack E RTU over Ethernet via TCP/IP.

Select the Workbench *Deployment View* window from the *View -> Deployment View* menu, or from the *Solution Explorer* window as shown below.




*Deployment View Window*

From the *Deployment View* window, select the *Network* connection (shown above as circled). The *Properties* window should then show the available Workbench network settings (see below).




*TCP/IP Network Settings*

The ETCP network driver has one connection property. Enter the TCP/IP address of the SCADAPack E RTU target as shown above. The ETCP driver uses TCP port 1131 to connect to the RTU target.

Using Ethernet communications for Workbench connections requires the RTU has **ISaGRAF/TCP** service enabled (see SCADAPack E Configurator **TCP/IP** page)

### 9.2.2 TCP/IP Communications Server

SCADAPack Workbench communications to the SCADAPack E RTU's ISaGRAF TCP/IP communications server may be established via the RTU's Ethernet interface or a serial PPP interface.

This requires the RTU has **ISaGRAF/TCP** service enabled (see SCADAPack E Configurator **TCP/IP** page) as shown below.


*SCADAPack E Configurator Enabled TCP/IP Services*

## 9.3 Application File Transfer

IEC 61131-3 applications are stored as a ***set of files*** 47 in a non-volatile file system directory called "C:\ISaGRAF5" on the SCADAPack E Smart RTU. However, SCADAPack Workbench creates a single file containing the content for the RTU. This *Application Package* file is used by SCADAPack E Configurator and ClearSCADA to load applications using DNP3 or IEC 60870-5 file transfer, or by a user to transfer the file via FTP. The firmware unpacks the file and activates the IEC 61131-3 Resources.

An *Application Package* file is produced by the Workbench each time a project is successfully built and the Solution is in the correct form. A message is logged in the output window if the solution is not in the correct form. This output can be used to verify that the created project can be loaded in the SCADAPack E RTU.

The file extension of an *Application Package* file is ".I5P". The file name root is the same as the Workbench Device, e.g. "SCADAPack300E.I5P".

### SCADAPack E Configurator ISaGRAF Application Transfer

SCADAPack E Configurator provides a simple method for downloading and activating IEC 61131-3 applications to the RTU.

Use the  icon on SCADAPack E Configurator Toolbar and follow the dialog to select the appropriate *.I5P file. The SCADAPack E Configurator will transfer the selected package file to the RTU and activates the IEC 61131-3 Resources automatically.

**⚠WARNING**

**UNINTENDED EQUIPMENT OPERATION**

Evaluate the operational state of the equipment monitored and controlled by the RTU before downloading. Hazardous situations can occur if system state is not confirmed prior to downloading.

**Failure to follow these instructions can result in death, serious injury or equipment damage.**

### Manual Application Transfer

IEC 61131-3 applications may be loaded from a Workbench-built *.I5P *Application Package* file, into the RTU file system. These files can be found in the workstation's SCADAPack Workbench "Documents\SCADAPack Workbench\Solutions" directory tree. The file transfer may be performed using:

- DNP3 protocol - File transfer
- FTP – TCP/IP File transfer
- IEC60870-5 - File transfer

Details of performing this file transfer are provided in the *SCADAPack E Operational Reference* manual.

After transferring a new package file to the RTU, the following command-line command unpacks the resource files and restarts the IEC 61131-3 application.

```
RESTART ISAGRAF [filename]
```

The syntax of the RESTART ISAGRAF command is shown in the following table.

| Command | Action |
|---|---|
| RESTART ISaGRAF | Restart Resource 1 and Resource 2 from the existing C:\ISAGRAF5\ folder for **Target 5**. |
| RESTART ISaGRAF 1 | Restart Resource 1 from the existing C:\ISAGRAF5 \ folder for **Target 5**. |
| RESTART ISaGRAF 2 | Restart Resource 2 from the existing C:\ISAGRAF5 \ folder for **Target 5**. |
| RESTART ISaGRAF *filename* | Unpack project from the .I5P file into the "C: \ISAGRAF5" directory, and restart Resource 1 and Resource 2.<br>If a Resource is missing from the package, it removes that resource.<br>If filename is not found or the format is invalid, it generates a status code and returns.<br>'1' and '2' are invalid project filenames. |

The subsequent state of the IEC 61131-3 Resources can then be confirmed using the SCADAPack E Configurator's *Logic -> IEC61131-3* page or from the "Status" command-line command.

# 10 Troubleshooting & Status Codes

This section describes the different sources of IEC 61131-3 target status codes. These status codes are accessible from the SCADAPack Workbench debugger or SCADAPack E RTU diagnostics output or *System Status* point.



## 10.1 Status Descriptions

RTU *System Status* codes are generated from IEC 61131-3 Warning Messages into Analog System point 50020. (RTU status codes are in decimal in the range 4001-4099)

Status messages are shown in the RTU diagnostics output in the form:

```
ISaGRAF5>>Warning: 4011, Resource Stopping
```

Status values (8-digit hex values) appear in the Workbench System Event Log. The 8-digit value shown in the system event log is shown in two 4-digit groups in this table for clarity (e.g. 0x1202:1087). This is also consistent with the status code format presented below.

**Status codes presented by the IEC 61131-3 target**

| Status code | Workbench System Event Log Code | Description | Suggested Action |
|---|---|---|---|
| 4001 | 0x2202:0001 | ISaGRAF 5 Start-up unsuccessful | Contact Schneider Electric technical support |
| 4002* | 0x2202:0002 | ISaGRAF 5 Communication Accept unsuccessful | Retry workbench communication with SCADAPack E RTU IEC 61131-3 target |
| 4003 | 0x2202:0003 | ISaGRAF 5 Resource restore unsuccessful | No action is required. This feature is not used on the SCADAPack E Smart RTU. |
| 4004 | 0x2202:0004 | ISaGRAF 5 Retain initialization unsuccessful | Reload IEC 61131-3 application |
| 4005 | 0x2202:0005 | ISaGRAF 5 Retain bad memory description | Do not use workbench retain memory description for SCADAPack E RTU |
| 4006 | 0x2202:0006 | ISaGRAF 5 Retain CRC changed | No action is required. |
| 4007 | 0x2202:0007 | ISaGRAF 5 Read retain variables from backup unsuccessful | Check file system is not full, Reload IEC 61131-3 application |

| Status code | Workbench System Event Log Code | Description | Suggested Action |
|---|---|---|---|
| 4008 | 0x2202:0008 | ISaGRAF 5 Retain Write unsuccessful | Contact Schneider Electric technical support |
| 4009 | 0x2202:0009 | ISaGRAF 5 Resource data allocation unsuccessful | Reload IEC 61131-3 application |
| 4010* | 0x1202:000A | ISaGRAF 5 Resource Start Report | |
| 4011* | 0x1202:000B | ISaGRAF 5 Resource Stop Report | |
| 4012 | 0x2202:000C | ISaGRAF 5 Standard function not implemented | Function cannot be used in IEC 61131-3 application, remove from application |
| 4013 | 0x2202:000D | ISaGRAF 5 Standard function block initialise unsuccessful | Contact Schneider Electric technical support |
| 4014 | 0x2202:000E | ISaGRAF 5 Standard function block exit unsuccessful | Contact Schneider Electric technical support |
| 4015 | 0x2202:000F | ISaGRAF 5 Standard function block not implemented | Function block cannot be used in IEC 61131-3 application, remove from application |
| 4016 | 0x2202:0010 | ISaGRAF 5 Function not found on SCADAPack E RTU | Function not available in SCADAPack E. Use SCADAPack Workbench to Import latest target definition into application project, upgrade SCADAPack E firmware, remove from IEC 61131-3 application |
| 4017 | 0x2202:0011 | ISaGRAF 5 SCADAPack E function block initialise unsuccessful | Contact Schneider Electric technical support |
| 4018 | 0x2202:0012 | ISaGRAF 5 SCADAPack E function block exit unsuccessful | Contact Schneider Electric technical support |
| 4019 | 0x2202:0013 | ISaGRAF 5 Function block not found on SCADAPack E RTU | Function block not available in SCADAPack E RTU. Use SCADAPack Workbench to Import latest target definition into application project, upgrade SCADAPack E firmware, remove from IEC 61131-3 application |
| 4020 | 0x2202:0014 | ISaGRAF 5 Conversion definition not found on SCADAPack E | Conversion definition is not available in SCADAPack E RTU; remove from IEC 61131-3 application |

| Status code | Workbench System Event Log Code | Description | Suggested Action |
|---|---|---|---|
| 4021 | 0x2202:0015 | ISaGRAF 5 I/O device not found on SCADAPack E | I/O device is not available in SCADAPack E RTU. Use SCADAPack Workbench to Import latest target definition into application project, upgrade SCADAPack E firmware, remove from IEC 61131-3 application |
| 4022 | 0x2202:0016 | ISaGRAF 5 I/O driver initialise/exit internal was unsuccessful | Contact Schneider Electric technical support |
| 4023 | 0x2202:0017 | ISaGRAF 5 I/O device open/close internal was unsuccessful | Contact Schneider Electric technical support |
| 4024 | 0x2202:0018 | ISaGRAF 5 Device read internal was unsuccessful | Contact Schneider Electric technical support |
| 4025 | 0x2202:0019 | ISaGRAF 5 Device write internal was unsuccessful | Contact Schneider Electric technical support |
| 4026 | 0x2202:001A | ISaGRAF 5 Device I/O control internal was unsuccessful | Contact Schneider Electric technical support |
| 4027 | 0x2202:001B | ISaGRAF 5 I/O driver initialize was unsuccessful | Contact Schneider Electric technical support |
| 4028 | 0x2202:001C | ISaGRAF 5 I/O device open unsuccessful | Review SCADAPack E point configuration and IEC 61131-3 application I/O device settings |
| 4031 | 0x2202:001F | ISaGRAF 5 Unknown TIC code | Reload IEC 61131-3 application |
| 4032 | 0x2202:0020 | ISaGRAF 5 Unknown data type on conversion | Reload IEC 61131-3 application |
| 4033 | 0x2202:0021 | ISaGRAF 5 TIC boundary check (Array access out of range) | Review the IEC 61131-3 application |
| 4034 | 0x1202:0022 | ISaGRAF 5 SINT variable divided by zero | Review the IEC 61131-3 application |
| 4035 | 0x1202:0023 | ISaGRAF 5 DINT variable divided by zero | Review the IEC 61131-3 application |

| Status code | Workbench System Event Log Code | Description | Suggested Action |
|---|---|---|---|
| 4036 | 0x1202:0024 | ISaGRAF 5 REAL variable divided by zero | Review the IEC 61131-3 application |
| 4037 | 0x2202:0025 | ISaGRAF 5 Dynamic SFC behaviour processing unsuccessful | Review incorrect IEC 61131-3 application implementation |
| 4038 | 0x2202:0026 | ISaGRAF 5 Cannot Put action of specified SFC step in list of actions to execute | Contact Schneider Electric technical support |
| 4039* | 0x1202:0027 | ISaGRAF 5 Cycle Time Overflow | Increase the cycle time in the IEC 61131-3 application |
| 4040 | 0x2202:0028 | ISaGRAF 5 Instance of SFC function block not found | Function block not available in SCADAPack E RTU, remove from IEC 61131-3 application |
| 4041 | 0x1202:0029 | ISaGRAF 5 INT variable divided by zero | Review the IEC 61131-3 application |
| 4048 | 0x1202:0030 | ISaGRAF 5 LINT variable divided by zero | Review the IEC 61131-3 application |
| 4049 | 0x1202:0031 | ISaGRAF 5 USINT variable divided by zero | Review the IEC 61131-3 application |
| 4050 | 0x1202:0032 | ISaGRAF 5 UINT variable divided by zero | Review the IEC 61131-3 application |
| 4051 | 0x1202:0033 | ISaGRAF 5 UDINT variable divided by zero | Review the IEC 61131-3 application |
| 4052 | 0x1202:0034 | ISaGRAF 5 ULINT variable divided by zero | Review the IEC 61131-3 application |
| 4053 | 0x1202:0035 | ISaGRAF 5 LREAL variable divided by zero | Review the IEC 61131-3 application |

* These status codes will NOT generate an RTU System Status in system analog point 50020. Other listed IEC 61131-3 status codes DO update the RTU System Status.

## 10.2 Service Descriptions

Status codes reported by the IEC 61131-3 target in SCADAPack E RTU diagnostics are in Hexadecimal (hex) format.

```
ISaGRAF 5>> Error Code: 0x1234:5678
```

Detailed IEC 61131-3 target status codes are also visible in the SCADAPack Workbench using the System Event log. When the workbench is connected to the target in Debug mode, press the yellow lightning-bolt icon on the debug tool bar to start the system event logger (as shown below).

*Start System Events button*

The Workbench system event log output represents IEC 61131-3 status codes using 8 digit hexadecimal numbers.

It is easier to use the LAST 4 HEX DIGITS from the status code value when referencing the status codes in the table below. The RTU status code and the table below presents the 8 digit hex number in two groups of 4 digits for easier recognition.

These abbreviations are used in the table below.

- CGM = configuration manager task on the IEC 61131-3 target
- DSA = data sequential access
- ETCP = target TCP communication task for SCADAPack Workbench to RTU target communications
- Exchange dispatcher = resource to resource communication coordinator
- HSD = host system driver for resource to resource data exchange on the same device
- ISARSI = target serial communication task between the SCADAPack Workbench and RTU target
- IXL,IXS = exchange layers for resource to resource communication

**Service codes presented by the IEC 61131-3 target**

| IEC 61131-3 Service | Status Code (hex) | Description | Suggested Action |
|---|---|---|---|
| | 0x1201: 000A | Consumer binding variables unsuccessful | Contact Schneider Electric technical support |
| System Layer | 0x1202: 1000 | System pool allocation unsuccessful | Contact Schneider Electric technical support |
| | 0x1202: 1001 | Start-up unsuccessful. Too many initialisations | |

| IEC 61131-3 Service | Status Code (hex) | Description | Suggested Action |
|---|---|---|---|
| | | performed | |
| | 0x1202: 1002 | Bad owner number (too high) | |
| | 0x1202: 1003 | Bad user number (too high) | |
| | 0x1202: 1004 | Bad object number (too high) | |
| | 0x1202: 1010 | Space identifier is not valid | |
| | 0x1202: 1011 | Owner number is not available | |
| | 0x1202: 1012 | Unable to create space | |
| | 0x1202: 1013 | Space already exists | May occur as a result of Online Change |
| | 0x1202: 1014 | Unable to delete space | Contact Schneider Electric technical support |
| | 0x1202: 1015 | Space is deleted or cannot link with space | |
| | 0x1202: 1016 | Cannot unlink from the space | |
| | 0x1202: 1017 | Unable to save space | Check SCADAPack E RTU file system |
| | 0x1202: 1018 | Unable to load space | Reload the IEC 61131-3 application |
| | 0x1202: 1019 | Cannot load space | Reload the IEC 61131-3 application |
| | 0x1202: 101A | Cannot remove space | Check SCADAPack E RTU file system |
| | 0x1202: 1020 | Internal semaphore | Contact Schneider Electric technical support |
| | 0x1202: 1021 | Semaphore identifier is not valid | |
| | 0x1202: 1022 | Owner number is not available | |
| | 0x1202: 1023 | Cannot create semaphore | |
| | 0x1202: | Semaphore already exists | |

| IEC 61131-3 Service | Status Code (hex) | Description | Suggested Action |
|---|---|---|---|
| | 1024 | | |
| | 0x1202: 1025 | Cannot delete semaphore | May occur as a result of Online Change. No action required. |
| | 0x1202: 1026 | Linking semaphore unsuccessful | Contact Schneider Electric technical support |
| | 0x1202: 1027 | Unlinking semaphore unsuccessful | |
| | 0x1202: 1028 | Cannot take semaphore | |
| | 0x1202: 1029 | Semaphore timeout | |
| | 0x1202: 1030 | Releasing semaphore unsuccessful | |
| | 0x1202: 1031 | Cannot create message queue | |
| | 0x1202: 1032 | Message queue already exists | |
| | 0x1202: 1033 | Size of message queue is too long | |
| | 0x1202: 1034 | Message is too long | |
| | 0x1202: 1035 | Could not delete message queue | |
| | 0x1202: 1036 | Cannot open message queue | Restart communication with SCADAPack E RTU |
| | 0x1202: 1037 | Could not close message queue | Contact Schneider Electric technical support |
| | 0x1202: 1038 | Could not send message | Restart communication with SCADAPack E RTU |
| | 0x1202: 1039 | Time out is reached sending message | Restart communication with SCADAPack E RTU |
| | 0x1202: 103A | Message is too long | Contact Schneider Electric technical support |
| | 0x1202: 103B | Priority parameter is wrong | Contact Schneider Electric technical support |
| | 0x1202: 103C | Could not receive message | Restart communication with SCADAPack E RTU |

| IEC 61131-3 Service | Status Code (hex) | Description | Suggested Action |
|---|---|---|---|
| | 0x1202: 103D | Time out is reached receiving message | Communication status. No action required |
| | 0x1202: 103E | The message is discarded. The buffer is too small. | Contact Schneider Electric technical support |
| | 0x1202: 1060 | Invalid DSA name | Contact Schneider Electric technical support |
| | 0x1202: 1061 | Cannot open DSA | |
| | 0x1202: 1062 | Cannot remove DSA | |
| | 0x1202: 1063 | Cannot create DSA | |
| | 0x1202: 1064 | Cannot write DSA | |
| | 0x1202: 1065 | Cannot read DSA | |
| | 0x1202: 1070 | Task is not running | |
| | 0x1202: 1071 | Cannot create task | Retry the request, Restart the SCADAPack E RTU |
| | 0x1202: 1072 | Cannot terminate task | Contact Schneider Electric technical support |
| | 0x1202: 1080 | Cannot initialise sockets | Contact Schneider Electric technical support |
| | 0x1202: 1081 | Cannot create socket | Contact Schneider Electric technical support |
| | 0x1202: 1082 | Cannot bind a socket | Check SCADAPack E RTU TCP/IP configuration, contact Schneider Electric technical support |
| | 0x1202: 1083 | Cannot listen on a socket | Contact Schneider Electric technical support |
| | 0x1202: 1084 | Cannot accept a socket | Contact Schneider Electric technical support |
| | 0x1202: 1085 | Invalid address | Review the IEC 61131-3 application settings, check SCADAPack E RTU TCP/IP configuration |
| | 0x1202: | Cannot connect a socket | Check SCADAPack E RTU |

| IEC 61131-3 Service | Status Code (hex) | Description | Suggested Action |
|---|---|---|---|
| | 1086 | | TCP/IP configuration, contact Schneider Electric technical support |
| | 0x1202: 1087 | Connection broken | Reconnect communication with SCADAPack E RTU |
| | 0x1202: 1088 | Could not receive data from socket | Reconnect communication with SCADAPack E RTU, restart the RTU |
| | 0x1202: 1089 | Could not send data on socket | Reconnect communication with SCADAPack E RTU, restart the RTU |
| | 0x1202: 108A | Cannot set socket option | Contact Schneider Electric technical support |
| | 0x1202: 108B | Socket option not implemented | Contact Schneider Electric technical support |
| Kernel | 0x1202: 1100 | Target name mismatch | Review the IEC 61131-3 application |
| | 0x1202: 1101 | Generated code and configuration mismatch | Rebuild the IEC 61131-3 application and reload |
| | 0x1202: 1102 | Database CRC mismatch | Rebuild the IEC 61131-3 application and reload |
| | 0x1202: 1103 | Module name mismatch | Rebuild the IEC 61131-3 application and reload |
| | 0x1202: 1104 | Resource name mismatch | Rebuild the IEC 61131-3 application and reload |
| | 0x1202: 1105 | Corrupted module | Rebuild the IEC 61131-3 application and reload |
| | 0x1202: 1110 | Target segmentation mismatch | Review the IEC 61131-3 application, import a target definition matching the SCADAPack E firmware |
| | 0x1202: 1111 | Too many blocks of memory to allocate | Review the IEC 61131-3 application |
| | 0x1202: 1112 | System variables overlap | Review the IEC 61131-3 application, import a target definition matching the SCADAPack E firmware |
| | 0x1202: 1120 | Memory allocated is too short | Contact Schneider Electric technical support |
| | 0x1202: | Cannot load driver | Review the IEC 61131-3 |

| IEC 61131-3 Service | Status Code (hex) | Description | Suggested Action |
|---|---|---|---|
| | 1121 | | application |
| | 0x1202: 1122 | Driver is not loaded | Review the IEC 61131-3 application |
| | 0x1202: 1123 | Invalid driver | Contact Schneider Electric technical support |
| | 0x1202: 1130 | Online modification not initialised | Contact Schneider Electric technical support |
| | 0x1202: 1131 | Can't modify when online modify space size is zero | Review the IEC 61131-3 application, stop the resource and download the application |
| | 0x1202: 1132 | Not enough memory for online modifications | Review the IEC 61131-3 application, stop the resource and download the application |
| | 0x1202: 1133 | No new modifications to update | |
| | 0x1202: 1134 | Cannot update program (new objects within POU) | Stop the resource and download the application |
| | 0x1202: 1135 | Cannot save online modifications | Check the SCADAPack E RTU file system, stop the resource and download the application |
| | 0x1202: 1136 | Online changes are not allowed | Stop the resource and download the application |
| | 0x1202: 1140 | Could not initialize SFC function block | Review the IEC 61131-3 application |
| | 0x1202: 1141 | Space allocation unsuccessful when initialising SFC function block | Contact Schneider Electric technical support |
| | 0x1202: 1142 | SFC function block table corrupted | Reload the IEC 61131-3 application |
| | 0x1202: 1180 | Resource number not allowed | Review the IEC 61131-3 application (SCADAPack E RTU valid resource numbers are 1 and 2 only) |
| | 0x1202: 1181 | Kernel is not in the appropriate state | Retry the request, restart the SCADAPack E RTU |
| | 0x1202: | Bad parameters in request | Retry the request |

| IEC 61131-3 Service | Status Code (hex) | Description | Suggested Action |
|---|---|---|---|
| | 1182 | | |
| | 0x1202: 1200 | Cannot allocate server memory | Contact Schneider Electric technical support |
| | 0x1202: 1201 | Cannot create server connection message queue | |
| | 0x1202: 1202 | Server message buffer too small | |
| | 0x1202: 1203 | Cannot remove server connection | |
| | 0x1202: 1204 | No more server connections available | |
| | 0x1202: 1205 | Cannot link with client message queue | Retry the request |
| | 0x1202: 1206 | Invalid server connection | Retry the request |
| | 0x1202: 1207 | Server message too large, request discarded | Contact Schneider Electric technical support |
| | 0x1202: 1208 | Timeout in received server message | Retry the request |
| | 0x1202: 1209 | Server bad reply | Retry the request |
| CMG task | 0x1202: 1300 | Cannot start new kernel | Restart the SCADAPack E RTU |
| | 0x1202: 1301 | Kernel is already running | |
| | 0x1202: 1302 | Kernel is not running | Start the Resource, Restart the SCADAPack E RTU |
| | 0x1202: 1310 | Cannot start configuration manager task | Restart the SCADAPack E RTU |
| Exchange dispatcher | 0x1202: 1400 | Allocation unsuccessful | Restart the SCADAPack E RTU |
| | 0x1202: 1401 | Trying to connect to an unknown resource | Check the IEC 61131-3 application configuration |
| | 0x1202: 1402 | Network device is not loaded | Check the IEC 61131-3 application |
| | 0x1202: 1403 | Network device is not found | Reconnect communication with the SCADAPack E RTU, check the IEC 61131-3 application configuration |

| IEC 61131-3 Service | Status Code (hex) | Description | Suggested Action |
|---|---|---|---|
| | 0x1202: 1404 | Operation due to system status code | Restart the SCADAPack E RTU |
| | 0x1202: 1405 | Received data for a close connection | Check network configuration |
| | 0x1202: 1406 | Connection not available | Reconnect communication with SCADAPack E RTU |
| | 0x1202: 1407 | Bad connection identifier | Restart the SCADAPack E RTU, check the IEC 61131-3 application configuration |
| | 0x1202: 1408 | Too many pending messages | Check the network configuration, check the IEC 61131-3 application settings |
| | 0x1202: 1409 | Message overflow | Check the IEC 61131-3 application settings |
| ETCP task | 0x1202: 1410 | Bad channel identifier | Restart the SCADAPack E RTU |
| | 0x1202: 1411 | Channel table full | Restart the SCADAPack E RTU |
| | 0x1202: 1412 | Connection refused | Restart the SCADAPack E RTU |
| | 0x1202: 1413 | Operation has no effect | Restart the SCADAPack E RTU |
| | 0x1202: 1414 | Attempt to access to closed socket | Restart the SCADAPack E RTU |
| | 0x1202: 1415 | No IXD to accept connection | Restart the SCADAPack E RTU |
| | 0x1202: 1416 | Connection refused (out of space) | Restart the SCADAPack E RTU |
| | 0x1202: 1417 | Bad parameter related to channel operation | Check IEC 61131-3 application connection parameters |
| | 0x1202: 1418 | Target communication server overloaded | Retry later, restart the SCADAPack E RTU |
| Common | 0x1202: 1420 | Binding unsuccessful. Common data exchange space full | Review the IEC 61131-3 application |
| | 0x1202: 1421 | Cannot link to data exchange producer | Review the IEC 61131-3 application, check the network connection |

| IEC 61131-3 Service | Status Code (hex) | Description | Suggested Action |
|---|---|---|---|
| | 0x1202: 1422 | Bad binding parameter | Review the IEC 61131-3 application |
| | 0x1202: 1430 | There is no network device table | Review the IEC 61131-3 application |
| | 0x1202: 1431 | Resource not found | Review the IEC 61131-3 application |
| | 0x1202: 1432 | Variable conversion unsuccessful | Review the IEC 61131-3 application |
| Exchange layer | 0x1202: 1500 | Memory block allocated for device is too short | Review the IEC 61131-3 applications |
| | 0x1202: 1501 | Cannot establish connection | Review networking configurations |
| | 0x1202: 1502 | Cannot remove connection | Restart the SCADAPack E RTU |
| | 0x1202: 1503 | Cannot read variables | Rebuild and reload IEC 61131-3 application |
| | 0x1202: 1504 | Too late to change device (connection may already be established) | |
| | 0x1202: 1505 | Cannot set device parameters | Review the IEC 61131-3 application |
| | 0x1202: 1506 | Memory block allocated for connection is too short | Contact Schneider Electric technical support |
| | 0x1202: 1507 | Exchange layer timeout | Review the IEC 61131-3 application settings |
| | 0x1202: 1508 | Exchange transport unsuccessful | Review network connections & configurations |
| | 0x1202: 1509 | The request code does not correspond | Contact Schneider Electric technical support |
| | 0x1202: 150A | The maximum capacity of the buffer is reached | Contact Schneider Electric technical support |
| | 0x1202: 150B | The notification identifier is wrong | Retry the request |
| | 0x1202: 150C | Bad return check during the transport | Retry the request |
| | 0x1202: 150D | Cannot remove connection | Contact Schneider Electric technical support |
| | 0x1202: | This function required a | Contact Schneider Electric |

| IEC 61131-3 Service | Status Code (hex) | Description | Suggested Action |
|---|---|---|---|
| | 150E | header for the buffer | technical support |
| | 0x1202: 150F | Unknown type | Contact Schneider Electric technical support |
| | 0x1202: 1510 | Bad index number | Contact Schneider Electric technical support |
| | 0x1202: 1511 | Start dialog is not allowed (maybe dialog is already established) | Retry the request |
| | 0x1202: 1512 | Stop dialog is not allowed | Retry the request |
| | 0x1202: 1513 | Start dialog procedure not completed | Restart communications |
| | 0x1202: 1514 | Stop dialog procedure not completed | Restart communications |
| | 0x1202: 1515 | Start not in progress | Restart communications |
| | 0x1202: 1516 | Stop not in progress | Restart communications |
| | 0x1202: 1518 | Dialog is not established | Restart communications |
| | 0x1202: 1519 | Variable description | Contact Schneider Electric technical support |
| | 0x1202: 151A | Method not provided by the driver or invalid method | Contact Schneider Electric technical support |
| | 0x1202: 151B | Service not provided by the driver or invalid service | Contact Schneider Electric technical support |
| | 0x1202: 151C | Size allowed for this variable is too short | Contact Schneider Electric technical support |
| | 0x1202: 151D | Only one connection by resource and by method allowed | Review the IEC 61131-3 application |
| | 0x1202: 151F | Bad extra parameters for connection | Review the IEC 61131-3 application |
| | 0x1202: 1520 | Request cannot be sent, retry later | Review the IEC 61131-3 application |
| | 0x1202: 1521 | Cannot establish connection, no more free exchange connections | Review the IEC 61131-3 application |
| | 0x1202: | Exchange layer internal | Contact Schneider Electric |

| IEC 61131-3 Service | Status Code (hex) | Description | Suggested Action |
|---|---|---|---|
| | 1522 | | technical support |
| | 0x1202: 1523 | IXL Cannot write variables | Contact Schneider Electric technical support |
| | 0x1202: 1530 | Invalid IXL identifier | Contact Schneider Electric technical support |
| | 0x1202: 1531 | Too many IXL initialisation loops | Restart SCADAPack E RTU |
| | 0x1202: 1532 | Clients cannot have the same number | Review the IEC 61131-3 application |
| | 0x1202: 1533 | Device registration successful | |
| | 0x1202: 1534 | Invalid driver name. Cannot register | Review the IEC 61131-3 application |
| | 0x1202: 1535 | Cannot register driver due to null parameter | Review the IEC 61131-3 application |
| | 0x1202: 1536 | Cannot register driver, maximum driver limit reached | Review the IEC 61131-3 application |
| | 0x1202: 1537 | Not all drivers configured. ISaRSI driver cannot be configured | Review the IEC 61131-3 application |
| | 0x1202: 1538 | Invalid connection identifier | Review the IEC 61131-3 application |
| | 0x1202: 1539 | Cannot establish connection, maximum connections reached | Restart communications |
| | 0x1202: 153A | Connection not established. Unknown driver | Review the IEC 61131-3 application |
| | 0x1202: 153B | Message overflow | Review the IEC 61131-3 application |
| | 0x1202: 153C | Exchange capability not implemented | Review the IEC 61131-3 application |
| | 0x1202: 153D | Exchange parameters are bad | Review the IEC 61131-3 application |
| | 0x1202: 153E | Bad Exchange request | Retry the request |
| | 0x1202: 153F | Kernel unsuccessful in executing request | Retry the request (commonly occurs after a |

| IEC 61131-3 Service | Status Code (hex) | Description | Suggested Action |
|---|---|---|---|
| | | | communication disconnection) |
| | 0x1202: 1540 | Symbol Table is not loaded | Rebuild and reload the IEC 61131-3 application |
| | 0x1202: 1541 | IXL: Maximum iteration is reached in symbol management | Contact Schneider Electric technical support |
| | 0x1202: 1542 | IXL: Variable is unknown | Contact Schneider Electric technical support |
| | 0x1202: 1543 | IXL: Type or Sub-type is unknown | Contact Schneider Electric technical support |
| | 0x1202: 1544 | IXL: Symbols mismatch | Reload the IEC 61131-3 application |
| | 0x1202: 1545 | IXL: Symbols mismatch, bad CRC | Reload the IEC 61131-3 application |
| | 0x1202: 1546 | IXL: Symbols mismatch, bad resource name | Reload the IEC 61131-3 application |
| | 0x1202: 1547 | IXL: End of symbols is reached or stop is required | Reload the IEC 61131-3 application |
| | 0x1202: 1548 | IXL: Symbols are corrupted | Reload the IEC 61131-3 application |
| | 0x1202: 1549 | IXL: Symbols are already loaded | |
| | 0x1202: 154A | IXL: Symbols are currently loading | |
| | 0x1202: 154B | IXL: Both IXL symbol version mismatch | Rebuild and reload the IEC 61131-3 application |
| | 0x1202: 154C | IXL: Device is unknown | Contact Schneider Electric technical support |
| | 0x1202: 154D | IXL: Syntax | Reload the IEC 61131-3 application |
| | 0x1202: 154E | IXL: Symbols table is incomplete, it is reduced one | Reload the IEC 61131-3 application, Review the IEC 61131-3 application |
| HSD driver | 0x1202: 1630 | Incompatible version of binding table | Rebuild and reload the IEC 61131-3 application |
| | 0x1202: 1631 | Produced variables refresh timeout | Review the IEC 61131-3 application, check network connection |

| IEC 61131-3 Service | Status Code (hex) | Description | Suggested Action |
|---|---|---|---|
| | 0x1202: 1632 | There is no producer | Review the IEC 61131-3 applications, check network connection |
| | 0x1202: 1633 | HSD binding service is not implemented | Review the IEC 61131-3 applications, don't use the resource binding features |
| ISaRSI task | 0x1202: 1640 | Cannot initialise serial device | Check SCADAPack E RTU port settings, contact Schneider Electric technical support |
| | 0x1202: 1641 | Cannot open serial device | Check SCADAPack E RTU port settings, contact Schneider Electric technical support |
| | 0x1202: 1642 | Cannot read serial device | Contact Schneider Electric technical support |
| | 0x1202: 1643 | Cannot write serial device | Contact Schneider Electric technical support |
| | 0x1202: 1644 | Bad parameters | Check IEC 61131-3 application connection parameters |
| ETCP task | 0x1202: 1700 | Create socket was unsuccessful | Restart the SCADAPack E RTU |
| | 0x1202: 1701 | Close socket was unsuccessful | Restart the SCADAPack E RTU |
| | 0x1202: 1702 | Launching communication server was unsuccessful | Restart the SCADAPack E RTU |
| | 0x1202: 1703 | Connecting to remote node was unsuccessful | Check SCADAPack E TCP/IP configuration, check IEC 61131-3 application settings |
| | 0x1202: 1704 | Can't read from socket | Check network connection, restart the SCADAPack E RTU |
| | 0x1202: 1705 | Accepting remote client connect was unsuccessful | Check network connection, restart the SCADAPack E RTU |
| | 0x1202: 1706 | Initialization of the TCP/IP stack was unsuccessful | Restart the SCADAPack E RTU |
| | 0x1202: 1707 | Changing the socket status was unsuccessful | Restart the SCADAPack E RTU |

| IEC 61131-3 Service | Status Code (hex) | Description | Suggested Action |
|---|---|---|---|
| | 0x1202: 1708 | Broken connection | Reconnect communication with SCADAPack E RTU, restart the RTU |
| | 0x1202: 1709 | Socket write was unsuccessful | Check network connection, restart the SCADAPack E RTU |
| | 0x1202: 170A | Received data is not coherent | Reconnect workbench communication with SCADAPack E RTU |
| | 0x1202: 170B | Remote ETCP connection unsuccessful | Reconnect communications with SCADAPack E RTU |
| | 0x1202: 1710 | ETCP communication timed out | Reconnect communications with SCADAPack E RTU, check communication settings in C: \ProgramData\ISaGRAF\ISaGRAF Gateway\ OpcConfig.XML [Windows 7 and later] C:\Documents and Settings\All Users\Application Data\ISaGRAF\ISaGRAF Gateway\ OpcConfig.XML [Windows XP] |
| | 0x1202: 1711 | ETCP server is already connected to a default queue | Restart the SCADAPack E RTU |
| | 0x1202: 1712 | ETCP server is full | Reconnect workbench communication with SCADAPack E RTU, restart the RTU |
| | 0x1202: 1730 | Host address not resolved | Review the IEC 61131-3 application |
| | 0x1202: 1731 | Remote resource not found | Review the IEC 61131-3 applications, check network connection |
| | 0x1202: 1732 | ETCP is not running | Check SCADAPack E RTU TCP/IP services has IEC 61131-3 /TCP enabled |

| IEC 61131-3 Service | Status Code (hex) | Description | Suggested Action |
|---|---|---|---|
| | 0x1202: 1733 | Resource has no variables to bind | Review the IEC 61131-3 application |
| | 0x1202: 1734 | ETCP binding Service is not implemented | Review the IEC 61131-3 applications, don't use the resource binding features |
| | 0x1202: 1735 | Variable not found by ETCP | Review the IEC 61131-3 application |
| Operating Sys | 0x1202: 2001 | Creation of system semaphore was unsuccessful | Contact Schneider Electric technical support |
| | 0x1202: 2002 | System access timeout | Retry the operation, restart the SCADAPack E RTU, |
| | 0x1202: 2003 | System storage not supported | Contact Schneider Electric technical support |
| ETCP binding | 0x1204: 1009 | ETCP binding producer timeout | Check network connection |
| | 0x1204: 100A | ETCP binding consumer timeout | Check network connection |
| HSD binding | 0x1204: 2002 | HSD binding consumer unsuccessful | Contact Schneider Electric technical support |
| | 0x1204: 2003 | HSD binding producer unsuccessful | Contact Schneider Electric technical support |
| System layer | 0x1205: 0001 | Invalid number of events | Contact Schneider Electric technical support |
| | 0x1205: 0002 | Creation of system semaphore unsuccessful | |
| | 0x1205: 0003 | Opening system semaphore unsuccessful | |
| | 0x1205: 0004 | Taking system semaphore unsuccessful | |
| | 0x1205: 0005 | Creating system space unsuccessful | |
| | 0x1205: 0006 | Deleting system space unsuccessful | |
| | 0x1205: 0007 | Linking system space unsuccessful | |
| | 0x1205: 0008 | Unlinking system space unsuccessful | |
| | 0x1205: | Opening system space | |

| IEC 61131-3 Service | Status Code (hex) | Description | Suggested Action |
|---|---|---|---|
| | 0009 | unsuccessful | |
| CMG | 0x1206: 0001 | Starting ETCP task was unsuccessful | Contact Schneider Electric technical support |
| | 0x1206: 0002 | Starting ISARSI task was unsuccessful | |
| | 0x1206: 0003 | Starting ISaGRAF kernel was unsuccessful | |
| | 0x1206: 0004 | Starting Fieldbus task was unsuccessful | |
| Kernel init. | 0x2201: 0001 | Initialising Kernel block was unsuccessful | Contact Schneider Electric technical support |
| | 0x2201: 0002 | Allocation Kernel memory was unsuccessful | |
| | 0x2201: 0003 | Initialising standard functions was unsuccessful | |
| | 0x2201: 0004 | Initialising SCADAPack E functions was unsuccessful | |
| | 0x2201: 0005 | Initialising standard function blocks was unsuccessful | |
| | 0x2201: 0006 | Initialising SCADAPack E function blocks was unsuccessful | |
| | 0x2201: 0007 | Initialising conversions was unsuccessful | |
| | 0x2201: 0008 | Initialising I/O devices was unsuccessful | Check the SCADAPack E point configurations, review the IEC 61131-3 application I/O device parameters |
| | 0x2201: 0009 | Initialising binding was unsuccessful | Contact Schneider Electric technical support |
| | 0x2201: 000B | Initialising step-by-step debugger was unsuccessful | Contact Schneider Electric technical support |
| ETCP binding | 0x2204: 1001 | Link binding lock unsuccessful | Contact Schneider Electric technical support |
| | 0x2204: 1002 | Allocate export binding space unsuccessful | |
| | 0x2204: 1003 | Create export binding semaphore unsuccessful | |

| IEC 61131-3 Service | Status Code (hex) | Description | Suggested Action |
|---|---|---|---|
| | 0x2204: 1004 | Allocate import binding space unsuccessful | |
| | 0x2204: 1005 | Create import binding semaphore unsuccessful | |
| | 0x2204: 1007 | Register producer unsuccessful | |
| | 0x2204: 1008 | Register consumer unsuccessful | |
| | 0x2204: 1009 | Link binding variables unsuccessful | |
| HSD binding | 0x2204: 2001 | Link binding lock was unsuccessful | Contact Schneider Electric technical support |
| | 0x2204: 2002 | Link binding variables was unsuccessful | Contact Schneider Electric technical support |
| | 0x2206: 0004 | Resource routing table | Review IEC 61131-3 application resource binding |
| Exchange | 0x2207: 001 | IXL initialisation unsuccessful | Contact Schneider Electric technical support |
| | 0x2207: 002 | IXS initialisation unsuccessful | Contact Schneider Electric technical support |

# 11 Target Upgrade

A firmware upgrade on the SCADAPack E Smart RTU may add new functionality. If new firmware has additional I/O device definitions, function blocks and capabilities the Target Definition must be updated in existing projects, or the resources will not start when downloaded.

SCADAPack Workbench can import the I/O device definitions, function blocks and capabilities of a target device from a *Target Definition* file (".tdb"). If new capabilities or features are added to the SCADAPack E firmware, a new set of .tdb files will be published and released as part of an updated SCADAPack Workbench release. New Workbench projects that are created from a ***Project Template*** 12 will not require a Target Definition file to be imported.

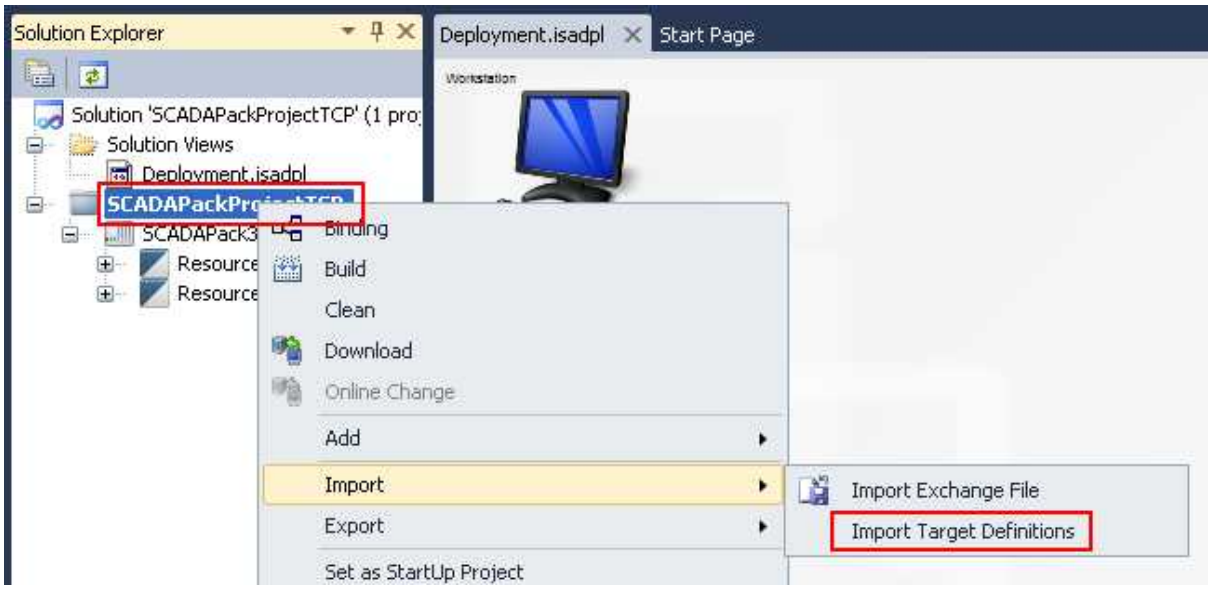
There are Two Target Definition files maintained for the SCADAPack Workbench. The files are named as follows:

- "SP_300E_Target_*.tdb" for the SCADAPack 300E RTU models, where * is the firmware version
- "SP_ES_Target_*.tdb" for the SCADAPack ES/ER RTU models, where * is the firmware version

To import a Target Definition file into an existing project:

1. From the SCADAPack Workbench *Solution Explorer* window, right-click the project and choose *Import*, then *Import Target Definitions* from the contextual menu (as shown below).
2. In the Open window, browse to locate the appropriate target definitions (*.tdb) file for your target to import into the project, then click Open. These files can be found in the workstation's SCADAPack Workbench "C:\Program Files\Schneider Electric\SCADAPack Workbench\SCADAPack Workbench\TargetDefinitions" directory tree.

When the importation process is completed, the features from the target definition are available for use in the project.




*Importing a 'Target Definitions' file*

# 12 Unsupported Features

## ISaGRAF 6.1 Unsupported Features

This section outlines the features that are documented in the ISaGRAF 6.1 Workbench User's Guide or ISaGRAF Target User's Guide that are *not* currently implemented in the SCADAPack E RTU Target.

Most other major ISaGRAF 6.1 features are implemented.

A target error will usually be produced if an IEC 61131-3 application or SCADAPack Workbench accesses an unimplemented feature. The unsupported features are as follows:

- Producer/Consumer Bindings between devices
- Interrupts feature
- Wiring on complex variable members
- *User System* Variables
- Scientific Apparatus Makers Association (SAMA) diagram language
- IEC 61499 language
- Failover Mechanism
- I/O Device Conversion, Gain, Offest
- SAFEBOOL variable types

## Target 3 Unsupported Features

This section documents *Target 3* features that are unsupported in *Target 5* or where a different mechanism is used in *Target 5*. This may be useful when porting existing *Target 3* applications using 16-bit Workbench (not covered by this document) and *Target 5* with 32-bit SCADAPack Workbench.

| Target 3 Feature | Suggested Action |
|---|---|
| "rea_msg" function | Replace with built-in function ANY_TO_STRING |
| "msg_ip" | Replace with STRING_TO_IP |
| "OPERATE" function | Replace with the appropriate SETPNTxx function |
| Conversion Tables | Not currently supported on Target 5. Replace with user-written Functions. |