SoMachine
SqlRemoteAccess Library Guide

06/2017

EIO0000002411.01

www.schneider-electric.com

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.

You agree not to reproduce, other than for your own personal, noncommercial use, all or part of this document on any medium whatsoever without permission of Schneider Electric, given in writing. You also agree not to establish any hypertext links to this document or its content. Schneider Electric does not grant any right or license for the personal and noncommercial use of the document or its content, except for a non-exclusive license to consult it on an "as is" basis, at your own risk. All other rights are reserved.

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.

Table of Contents

Safety Information 5
About the Book 9
Part I General Information 11
Chapter 1 Specific Safety Information 13
Qualification of Personnel 14
Proper Use 14
Product Related Information 15
Chapter 2 Presentation of the Library 19
General Information 19
Chapter 3 How to Transfer Data Via SQL Gateway Between SoMachine and Any Database 23
How Does It Work? 23
Part II Enumerations and Structures 27
Chapter 4 Enumerations 29
ET_Result 29
Chapter 5 Structures 33
ST_ConnectionSettings 33
Chapter 6 ALIAS (DUT) 35
ALIAS - Device Unit Types (DUT) 35
Part III Global Variables 37
Chapter 7 Global Constants List 39
Global Constants List (GCL) 39
Chapter 8 Global Parameter List 41
Global Parameter List (GPL) 41
Part IV Program Organization Units (POU) 43
Chapter 9 Function Blocks 45
FB_SqlDbRead 46
FB_SqlDbWrite 49
Chapter 10 Functions 51
FC_EtResultToString 51
Appendices 53

Appendix A Function and Function Block Representation 55
Differences Between a Function and a Function Block 56
How to Use a Function or a Function Block in IL Language 57
How to Use a Function or a Function Block in ST Language. 61
Glossary 65
Index 69

Safety Information

Important Information

NOTICE

Read these instructions carefully, and look at the equipment to become familiar with the device before trying to install, operate, service, 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 a hazardous situation which, if not avoided, **will result in** death or serious injury.

⚠ WARNING

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

⚠ CAUTION

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

NOTICE

NOTICE is used to address practices not related to physical injury.

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 its installation, and has received safety training to recognize and avoid the hazards involved.

BEFORE YOU BEGIN

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.

⚠ WARNING

UNGUARDED EQUIPMENT

- Do not use this software and related automation equipment on equipment which does not have point-of-operation protection.
- Do not reach into machinery during operation.

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

This automation equipment and related software is used to control a variety of industrial processes. The type or model of automation equipment suitable for each application will vary depending on factors such as the control function required, degree of protection required, production methods, unusual conditions, government regulations, etc. In some applications, more than one processor may be required, as when backup redundancy is needed.

Only you, the user, machine builder or system integrator can be aware of all the conditions and factors present during setup, operation, and maintenance of the machine and, therefore, can determine the automation equipment and the related safeties and interlocks which can be properly used. When selecting automation and control equipment and related software for a particular application, you should refer to the applicable local and national standards and regulations. The National Safety Council's Accident Prevention Manual (nationally recognized in the United States of America) also provides much useful information.

In some applications, such as packaging machinery, additional operator protection such as point-of-operation guarding must be provided. This is necessary if the operator's hands and other parts of the body are free to enter the pinch points or other hazardous areas and serious injury can occur. Software products alone cannot protect an operator from injury. For this reason the software cannot be substituted for or take the place of point-of-operation protection.

Ensure that appropriate safeties and mechanical/electrical interlocks related to point-of-operation protection have been installed and are operational before placing the equipment into service. All interlocks and safeties related to point-of-operation protection must be coordinated with the related automation equipment and software programming.

NOTE: Coordination of safeties and mechanical/electrical interlocks for point-of-operation protection is outside the scope of the Function Block Library, System User Guide, or other implementation referenced in this documentation.

START-UP AND TEST

Before using electrical control and automation equipment for regular operation after installation, the system should be given a start-up test by qualified personnel to verify correct operation of the equipment. It is important that arrangements for such a check be made and that enough time is allowed to perform complete and satisfactory testing.

⚠ WARNING

EQUIPMENT OPERATION HAZARD

- Verify that all installation and set up procedures have been completed.
- Before operational tests are performed, remove all blocks or other temporary holding means used for shipment from all component devices.
- Remove tools, meters, and debris from equipment.

Failure to follow these instructions can result in death, serious 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 temporary grounds that are not 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 all temporary grounds 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.

About the Book

At a Glance

Document Scope

This document describes the library SqlRemoteAccess.

The library provides SQL (Structured Query Language) client function blocks that allow your controller to connect to an SQL database in order to run SQL queries for reading and writing data.

As a prerequisite for using SQL functions, you have to install the SQL Gateway that is supplied with SoMachine as an optional component and that requires a specific license.

The SqlRemoteAccess library uses system functions and resources which are supported on specific controller platforms:

- Modicon M241 Logic Controller
- Modicon M251 Logic Controller
- Modicon M258 Logic Controller
- Modicon LMC078 Motion Controller
- Modicon LMC058 Motion Controller

Validity Note

This document has been updated for the release of SoMachine V4.3.

The technical characteristics of the devices described in this document also appear online. To access this information online:

Step	Action
1	Go to the Schneider Electric home page *www.schneider-electric.com*.
2	In the **Search** box type the reference of a product or the name of a product range. ● Do not include blank spaces in the reference or product range. ● To get information on grouping similar modules, use asterisks (*).
3	If you entered a reference, go to the **Product Datasheets** search results and click on the reference that interests you. If you entered the name of a product range, go to the **Product Ranges** search results and click on the product range that interests you.
4	If more than one reference appears in the **Products** search results, click on the reference that interests you.
5	Depending on the size of your screen, you may need to scroll down to see the data sheet.
6	To save or print a data sheet as a .pdf file, click **Download XXX product datasheet**.

The characteristics that are presented in this manual should be the same as those characteristics that appear online. In line with our policy of constant improvement, we may revise content over time to improve clarity and accuracy. If you see a difference between the manual and online information, use the online information as your reference.

Related Documents

Document title	Reference
SoMachine Functions and Libraries User Guide	*EIO0000000735 (ENG)*; *EIO0000000792 (FRE)*; *EIO0000000793 (GER)*; *EIO0000000795 (SPA)*; *EIO0000000794 (ITA)*; *EIO0000000796 (CHS)*
SoMachine Programming Guide	*EIO0000000067 (ENG)*; *EIO0000000069 (FRE)*; *EIO0000000068 (GER)*; *EIO0000000071 (SPA)*; *EIO0000000070 (ITA)*; *EIO0000000072 (CHS)*
SQL Gateway User Guide	*EIO0000002417 (ENG)*; *EIO0000002418 (FRE)*; *EIO0000002419 (GER)*; *EIO0000002421 (SPA)*; *EIO0000002420 (ITA)*; *EIO0000002422 (CHS)*

You can download these technical publications and other technical information from our website at *http://www.schneider-electric.com/en/download*.

Part I
General Information

What Is in This Part?

This part contains the following chapters:

Chapter	Chapter Name	Page
1	Specific Safety Information	13
2	Presentation of the Library	19
3	How to Transfer Data Via SQL Gateway Between SoMachine and Any Database	23

Chapter 1
Specific Safety Information

Overview

This section contains information regarding working with the SqlRemoteAccess library. Personnel working with the SqlRemoteAccess library must read and observe this information.

What Is in This Chapter?

This chapter contains the following topics:

Topic	Page
Qualification of Personnel	14
Proper Use	14
Product Related Information	15

Qualification of Personnel

Overview

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-related training to recognize and avoid the hazards involved.

The qualified person must be able to detect possible hazards that may arise from parameterization, modifying parameter values and generally from mechanical, electrical, or electronic equipment. The qualified person must be familiar with the standards, provisions, and regulations for the prevention of industrial accidents, which they must observe when designing and implementing the system.

Proper Use

Overview

This product is a library to be used together with the control systems and servo amplifiers intended solely for the purposes as described in the present documentation as applied in the industrial sector.

Always observe the applicable safety-related instructions, the specified conditions, and the technical data.

Perform a risk evaluation concerning the specific use before using the product. Take protective measures according to the result.

Since the product is used as a part of an overall system, you must ensure the safety of the personnel by means of the concept of this overall system (for example, machine concept).

Any other use is not intended and may be hazardous. Electrical devices and equipment must only be installed, operated, maintained, and repaired by qualified personnel.

Product Related Information

Product Related Information

⚠ WARNING

LOSS OF CONTROL

- The designer of any control scheme must consider the potential failure modes of control paths and, for certain critical control functions, provide a means to achieve a safe state during and after a path failure. Examples of critical control functions are emergency stop and overtravel stop, power outage and restart.
- Separate or redundant control paths must be provided for critical control functions.
- System control paths may include communication links. Consideration must be given to the implications of unanticipated transmission delays or failures of the link.
- Observe all accident prevention regulations and local safety guidelines.[1]
- Each implementation of this equipment must be individually and thoroughly tested for proper operation before being placed into service.

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

[1] For additional information, refer to NEMA ICS 1.1 (latest edition), "Safety Guidelines for the Application, Installation, and Maintenance of Solid State Control" and to NEMA ICS 7.1 (latest edition), "Safety Standards for Construction and Guide for Selection, Installation and Operation of Adjustable-Speed Drive Systems" or their equivalent governing your particular location.

Before you attempt to provide a solution (machine or process) for a specific application using the POUs found in the library, you must consider, conduct and complete best practices. These practices include, but are not limited to, risk analysis, functional safety, component compatibility, testing and system validation as they relate to this library.

⚠ WARNING

IMPROPER USE OF POUS

- Perform a safety-related analysis for the application and the devices installed.
- Ensure that the POUs are compatible with the devices in the system and have no unintended effects on the proper functioning of the system.
- Use appropriate parameters, especially limit values, and observe machine wear and stop behavior.
- Verify that the sensors and actuators are compatible with the selected POUs.
- Thoroughly test all functions during verification and commissioning in all operation modes.
- Provide independent methods for critical control functions (emergency stop, conditions for limit values being exceeded, etc.) according to a safety-related analysis, respective rules, and regulations.

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

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Only use software approved by Schneider Electric for use with this equipment.
- Update your application program every time you change the physical hardware configuration.

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

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Update your application program as required, paying particular attention to I/O address adjustments, whenever you modify the hardware configuration.

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

Incomplete file transfers, such as data files, application files and/or firmware files, may have serious consequences for your machine or controller. If you remove power, or if there is a power outage or communication interruption during a file transfer, your machine may become inoperative, or your application may attempt to operate on a corrupted data file. If a an interruption occurs, reattempt the transfer. Be sure to include in your risk analysis the impact of corrupted data files.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION, DATA LOSS, OR FILE CORRUPTION

- Do not interrupt an ongoing data transfer.
- If the transfer is interrupted for any reason, re-initiate the transfer.
- Do not place your machine into service until the file transfer has completed successfully, unless you have accounted for corrupted files in your risk analysis and have taken appropriate steps to prevent any potentially serious consequences due to unsuccessful file transfers.

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

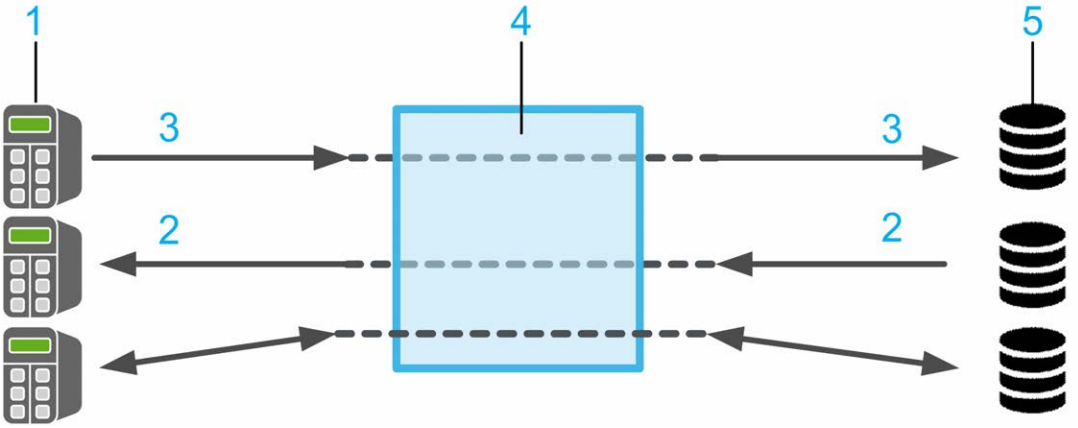
Chapter 2
Presentation of the Library

General Information

Introduction

The library SqlRemoteAccess provides SQL (Structured Query Language) client function blocks that allow your controller to connect to an SQL database in order to run SQL queries for reading and writing data.

The communication between the controller that acts as an SQL client and the SQL database server is running via the Schneider Electric SQL Gateway. Therefore, you have to install the SQL Gateway that is supplied with SoMachine as an optional component and that requires a specific license before you can use the SQL function. For further information, refer to the SQL Gateway User Guide *(see page 10)*.



1 1...n controllers (SQL clients)
2 Read data
3 Write data
4 SQL Gateway
5 1...n database servers

After successful installation, the controller can send a customized SQL query to the database server, for example:

- Querying data from tables.
- Inserting, changing, and deleting data in tables.
- Executing database procedures.

The following table indicates the characteristics of the library:

Characteristic	Value
Library title	SqlRemoteAccess
Company	Schneider Electric
Category	**Communication**
Component	**SQL Library**
Default namespace	**SE_SQL**
Language model attribute	Qualified-access-only *(see SoMachine, Functions and Libraries User Guide)*
Forward compatible library	Yes (FCL *(see SoMachine, Functions and Libraries User Guide)*)

NOTE: For this library, qualified-access-only is set. This means, that the POUs, data structures, enumerations, and constants have to be accessed using the namespace of the library. The default namespace of the library is **SE_SQL**.

General Considerations

Consider the following limitations for SQL communications:

- Only IPv4 (Internet Protocol version 4) is supported.
- Only database data types supported which conform to IEC 61131-3.
- Read and write BLOB (Binary Large Objects) objects from and into a database is not supported.

The library described in this document internally uses the TcpUdpCommunication library.

The TcpUdpCommunication (Schneider Electric) and the CAA Net Base Services library (CAA Technical Workgroup) use the same system resources on the controller. The simultaneous use of both libraries in the same application may lead to disturbances during the operation of the controller.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Do not use the library TcpUdpCommunication (Schneider Electric) together with the library CAA Net Base Services (CAA Technical Workgroup) simultaneously in the same application.

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

Considerations Concerning Cyber Security

The SqlRemoteAccess library functions do not support secure connections such as TLS (Transport Layer Security) or SSL (Secure Socket Layer) to the SQL Gateway. Communication must only be performed inside your industrial network, isolated from other networks inside your company, and protected from the Internet.

NOTE: Schneider Electric adheres to industry best practices in the development and implementation of control systems. This includes a "Defense-in-Depth" approach to secure an Industrial Control System. This approach places the controllers behind one or more firewalls to restrict access to authorized personnel and protocols only.

⚠ WARNING

UNAUTHENTICATED ACCESS AND SUBSEQUENT UNAUTHORIZED MACHINE OPERATION

- Evaluate whether your environment or your machines are connected to your critical infrastructure and, if so, take appropriate steps in terms of prevention, based on Defense-in-Depth, before connecting the automation system to any network.
- Limit the number of devices connected to a network to the minimum necessary.
- Isolate your industrial network from other networks inside your company.
- Protect any network against unintended access by using firewalls, VPN, or other, proven security measures.
- Monitor activities within your systems.
- Prevent subject devices from direct access or direct link by unauthorized parties or unauthenticated actions.
- Prepare a recovery plan including backup of your system and process information.

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

Overview of the POUs

Function block / function	Use
`FB_SqlDbRead` *(see page 46)*	Performs SQL requests that read data from the SQL database.
`FB_SqlDbWrite` *(see page 49)*	Performs SQL requests that update or modify the SQL database.
`FC_EtResultToString` *(see page 51)*	Converts an enumeration element of type `ET_Result` to a string variable.

Overview of the Structures in the Module-Specific Interface

Structure	Use
`ST_ConnectionSettings` *(see page 33)*	Contains the information for connecting to an SQL Gateway and information on the SQL database.

Overview of the Enumerations

Enumeration	Use
`ET_Result` *(see page 29)*	Contains the possible values that indicate the result of operations executed by the function block.

Chapter 3
How to Transfer Data Via SQL Gateway Between SoMachine and Any Database

How Does It Work?

System Requirements

The following system requirements are needed:

- Server (for example, MySQL, not supplied by Schneider Electric)
- SQL Client (for example, MySQL Workbench, not supplied by Schneider Electric)
- Database, created with the SQL Client (for example, MySQL Workbench)
- SQL Gateway (is a separate installation on SoMachine DVD)
- SqlRemoteAccess Library (is part of the SoMachine installation)

How to Transfer Data Via SQL Gateway from SoMachine to Any Database

Before data can be transferred, the SQL connection must be built with the SQL Gateway. For detailed information, refer to the SQL Gateway User Guide *(see page 10)*.

In order to interface with the SQL Gateway, you may need information about the SQL Gateway and about the database from the administrator of the SQL environment. You need the IP address of the SQL Gateway, the connection name configured in the SQL Gateway, and the name of the database.

NOTE: The SQL syntax in code examples in this description depends on a MySQL server, if you use another SQL server, the commands can be different.

Write Data from SoMachine to the Database

Declaration:

```
PROGRAM WriteData
VAR
     fbWrite           : SE_SQL.FB_SqlDbWrite;
     stConnSettings    : SE_SQL.ST_ConnectionSettings;
     refRequestWstring : SE_SQL.RequestWstring;
END_VAR
```

Implementation:

```
stConnSettings.wsDbConnectionName    := "ConnectionCountry";
stConnSettings.wGwPort               := 3457;
stConnSettings.wsGwIpAddress         := "10.128.154.47";
stConnSettings.timSqlTimeout         := T#20S;
//SQL Command to insert a new Country - MySQL-
Syntax: INSERT INTO <database name>.<table name> (column1...columnN) va
lues (value1...valueN)
//SQLite-Syntax: without "<database name>.")
refRequestWstring[1]                 := "INSERT INTO world.countries (C
ountry, CapitalCity, Language)";
refRequestWstring[2]                 := "values ('Germany', 'Berlin', '
german')";

fbWrite(    i_xExecute             := TRUE,
iq_stConnSettings                  := stConnSettings,
i_refRequestWstring                := refRequestWstring,
i_uiNumOfWstrings                  := 2
);
```

Description of the parameters used

Step	Action
1	In the ST_ConnectionSettings *(see page 33)*, set the IP address (wsGwIpAddress *(see page 33)*) of your PC where the gateway is running, for example, 10.128.154.47.
2	In the ST_ConnectionSettings *(see page 33)*, set the port (wGwPort *(see page 33)*) of the gateway (default: 3457).
3	In the ST_ConnectionSettings *(see page 33)*, set the connection name (wsDbConnectionName *(see page 33)*) which is configured in the SQL Gateway, for example, `ConnectionCountry`.
4	In the ST_ConnectionSettings *(see page 33)*, set the time-out (timSqlTimeout *(see page 33)*) for example, T#20S.
5	Write your request for writing in the refRequestWstring *(see page 46)*, for example, INSERT INTO world.countries (Country) values ('France').
6	In the i_uiNumOfWstrings *(see page 46)*, set your lines in the refRequestWstring *(see page 46)*, for example, 2.
7	Call the FB_SqlDbWrite *(see page 49)* with the settings/parameters/variables above.

NOTE: It is only possible to send one request per function block call.

Read Data from SoMachine to the Database

Declaration:

```
PROGRAM ReadData
VAR
        fbRead            : SE_SQL.FB_SqlDbRead;
        stConnSettings    : SE_SQL.ST_ConnectionSettings;
        refRequestWstring : SE_SQL.RequestWstring;
        refUserData       : SE_SQL.UserData;
END_VAR
```

Implementation:

```
stConnSettings.wsDbConnectionName  := "ConnectionCountry";
stConnSettings.wGwPort             := 3457;
stConnSettings.wsGwIpAddress       := "10.128.154.47";
stConnSettings.timSqlTimeout       := T#1 M;

//SQL Command to read all Countries in the database - MySQL-
Syntax: SELECT (column1...columnN) FROM <database name>.<table name> [w
here <column>=<value>]
//SQLite-Syntax: without "<database name>."
refRequestWstring[1]              := "SELECT * FROM world.countries";

fbRead  (      i_xExecute            := TRUE,
               iq_stConnSettings     := stConnSettings,
               i_refRequestWstring   := refRequestWstring,
               i_uiNumOfWstrings     := 1,
               i_refUserData         := refUserData
);
```

Description of the parameters used

Step	Action
1	In the ST_ConnectionSettings *(see page 33)*, set the IP address (wsGwIpAddress *(see page 33)*) of your PC where the SQL Gateway is running, for example, 10.128.154.47.
2	In the ST_ConnectionSettings *(see page 33)*, set the port (wGwPort *(see page 33)*) of the gateway (default: 3457).
3	In the ST_ConnectionSettings, *(see page 33)*, set the connection name (wsDbConnectionName *(see page 33)*) which is configured in the SQL Gateway, e.g `ConnectionCountry`.
4	In the ST_ConnectionSettings *(see page 33)*, set the time-out (timSqlTimeout *(see page 33)*), for example, T#20S.
5	Write your request for reading in the refRequestWstring *(see page 46)*, for example, `SELECT * FROM world.countries`.
6	In the i_uiNumOfWstrings *(see page 46)*, set your lines in the refRequestWstring *(see page 46)*, for example, 1.
7	In the i_refUserData *(see page 46)*, set your UserData *(see page 46)* Array.
8	Call the FB_SqlDbRead *(see page 46)* with the settings/parameters/variables above.

Advanced Settings for the Data Buffer / Request with the Global Parameters

NOTE: The advanced settings can be overwritten specifically for your project in the Library Manager.

Step	Action
1	In the Gc_uiMaxRows *(see page 42)*, set the maximum number of rows for the ARRAY of UserData (only needed for FB_SqlDbRead *(see page 46)*), for example, 20. **Result**: 20 rows from the configured database can be read.
2	In the Gc_uiMaxColumns *(see page 42)*, set the maximum number of columns for the ARRAY of UserData (only needed for FB_SqlDbRead), for example, 10. **Result**: 10 columns from the configured database can be read.
3	In the Gc_uiTableWstringLength *(see page 42)*, set the maximum number of characters in the 2-dimensional ARRAY of UserData (only needed for FB_SqlDbRead *(see page 46)*), for example, 200. **Result**: One field with 200 characters from the configured database can be read.
4	In the Gc_uiMaxRequest *(see page 42)*, set the maximum lines of the ARRAY of RequestWstring, for example, 20. **Result**: The request can be split into 20 lines of WSTRINGs.
5	In the Gc_uiRequestWstringLength *(see page 42)*, set the maximum number of characters in the ARRAY of RequestWstring, for example, 200. **Result**: One request line can include 200 characters.

Part II
Enumerations and Structures

What Is in This Part?

This part contains the following chapters:

Chapter	Chapter Name	Page
4	Enumerations	29
5	Structures	33
6	ALIAS (DUT)	35

Chapter 4
Enumerations

ET_Result

Overview

Type:	Enumeration
Available as of:	V1.0.0.0

Description

The enumeration `ET_Result` contains the possible values that indicate the result of operations executed by the function block.

Enumeration Elements

Name	Data type	Value	Description
If `q_xError` of a function block *(see page 45)* is FALSE, the following status message is shown.			
`Ok`	DINT	0	Operation completed successfully.
If `q_xError` of a function block *(see page 45)* is TRUE, one of the following status messages is shown.			
`TcpClientIssue`	DINT	1	The TCP operation was not completed successfully. General TCP issue, refer to the TcpUdpCommunication Library Guide *(see SoMachine, TcpUdpCommunication, Library Guide)*.
`InvalidIP`	DINT	26	The given IP address is not valid. Verify the setting of the `iq_stConnSettings.wsGpIpAddress` *(see page 33)* input.
`ClientNotConnected`	DINT	28	The client is not connected to the TCP server. Verify that the SQL Gateway is installed and running within the specified IP address and port.
`ConnectionTimedOut`	DINT	41	Remote system is not available or is not answering the request. Verify that the SQL Gateway is installed and running within the specified IP address and port.
`DbConnectionNameNull`	DINT	61	`DbConnectionName` is empty (Null). Element `wsDbConnectionName` from input `iq_stConnSettings` *(see page 33)* is blank.

Name	Data type	Value	Description
DataBufferTooSmall	DINT	63	User buffer is insufficient for returning data. Matching the returning numbers of rows and columns with the settings in the global parameter list *(see page 41)*, you can find the minimum settings at q_uiRetNumOfRows (q_uiRetNumOfColumns) of FB_SqlDbRead *(see page 46)* after each execution.
SqlDatabaseBusy	DINT	64	A former request is currently being processed. Another instance with the same input for iq_stConnSettings is currently executed (output q_xBusy = TRUE).
SqlGwToPlcCorruptedMessage	DINT	65	Controller received an invalid message from the SQL Gateway. Contact your local Schneider Electric support.
TimeoutExpired	DINT	66	Function block operation timeout expired. Verify the setting of the input iq_stConnSettings.timSqlTimeout *(see page 33)* depending on the system configuration the default value must be increased.
TransactionNumMismatch	DINT	67	Transaction number mismatch. Contact your local Schneider Electric support.
TransmissionNumMismatch	DINT	68	Transmission number mismatch. Contact your local Schneider Electric support.
ConnectionNotIdle	DINT	69	TcpUdp client connection already occupied with another service.Verify that no other instance of TCPUDP.FB_TCPClient inside your application occupies the connection.
SocketNotClosed	DINT	70	Method Shutdown or Close from TcpUdp client not successful. TCP connection issue. Restart your controller.
NoDataReturned	DINT	71	The function block FB_SqlDbGetData does not provide any data. The SQL request was executed successfully but no data returned. Verify that the assigned database contains data.
NotConnected	DINT	72	TcpUdp client system is not ready to receive or transmit data. TCP connection issue. Restart your controller.
InvalidPort	DINT	80	The given port number is equal to zero. Verify the iq_stConnSettings.wGwPort *(see page 33)* input. Usually, the default value 3457 is used.

Name	Data type	Value	Description
InvalidWstringNumber	DINT	81	The input i_uiNumOfWstrings is less than 1 or greater than GPL.Gc_uiMaxRequest. Verify the i_uiNumOfWstrings input. At least one WSTRING is required. Also verify that the given number is greater than the limit GPL.Gc_uiMaxRequest (GPL) *(see page 41)*.
InvalidRequestWstring	DINT	82	Empty request string or invalid reference (NULL). Verify the i_refRequestWstring *(see page 49)* input. The first WSTRING (i_refRequestWstring[1]) must contain any character. Verify that if the input is assigned.
InvalidUserDataWstring	DINT	83	Invalid user data reference (NULL). Verify that the i_refUserData *(see page 46)* input is assigned.
GplMaxRowsExceedsLimits	DINT	85	GPL.Gc_uiMaxRows is out of range. Verify the settings of Global Parameter List *(see page 41)*.
GplMaxColumnsExceedsLimits	DINT	86	GPL.Gc_uiMaxColumns is out of range. Verify the settings of Global Parameter List *(see page 41)*.
GplTableWstringExceedsLimits	DINT	87	GPL.Gc_uiTableWstringLength is out of range. Verify the settings of Global Parameter List *(see page 41)*.
GplRequestWstringExceedsLimits	DINT	88	GPL.Gc_uiRequestWstringLength is out of range. Verify the settings of Global Parameter List *(see page 41)*.
GplMaxRequestExceedsLimits	DINT	89	GPL.Gc_uiMaxRequest is out of range. Verify the settings of Global Parameter List *(see page 41)*.
GplMaxRecBufferExceedsLimits	DINT	91	GPL.Gc_uiRecBufferLength is out of range. Verify the settings of Global Parameter List *(see page 41)*.
GplMaxSendBufferExceedsLimits	DINT	92	GPL.Gc_uiSendBufferLength is out of range. Verify the settings of Global Parameter List *(see page 41)*.
ConnectionNameNotExisting	DINT	100	SQL Gateway connection name does not exist. Verify available database connections from SQL Gateway console.
DbConfigurationInvalid	DINT	101	SQL Gateway connection configuration does not match. Execute Test Database Connection from SQL Gateway console.
DbConnectionNotAvailable	DINT	102	SQL Gateway connection to the database does not work. Execute Test Database Connection from SQL Gateway console.

Name	Data type	Value	Description
SqlGwProtocolError	DINT	120	Protocol error between SQL library and gateway detected. Contact your local Schneider Electric support.
CommandExecutionIssue	DINT	150	Execution of the SQL request not successful. Verify i_refRequestWString *(see page 46)* input according to the required SQL syntax. Refer to SQL server manual.
NoQueryResultAvailable	DINT	151	SQL Gateway does not provide an answer. Verify that the addressed database contains any data with an additional application.
SqlGwInvalidLicense	DINT	170	SQL Gateway license issue. Open the License Manager to verify that **SQL for SoMachine** is valid.
SqlGwUnexpectedError	DINT	199	SQL Gateway error detected. Contact your local Schneider Electric support.

NOTE: The results in the range 100...199 are messages from SQL Gateway. Refer to SQL Gateway User Guide *(see page 10)*.

Used By

- FB_SqlDbRead
- FB_SqlDbWrite

Chapter 5
Structures

ST_ConnectionSettings

Overview

Type:	Structure
Available as of:	V1.0.0.0
Inherits from:	–

Description

The structure ST_ConnectionSettings contains the information for connecting to an SQL Gateway and information on the SQL database.

Structure Elements

Name	Data type	Description
wsDbConnectionName	WSTRING[40]	The value of this element must match the connection name configured in the SQL Gateway. Refer to *SQL Gateway User Guide* *(see page 10)*.
wsGwIpAddress	WSTRINGGPL.Gc_uiIpStringSize	The IP address of the SQL Gateway. This is the IP address of the server on which the gateway is running.
wGwPort	WORD	The port number of the SQL Gateway. Default value: 3457
timSqlTimeout	TIME	The timeout value that is applied to the SQL requests. Default value: 60s

Used By

- FB_SqlDbRead
- FB_SqlDbWrite

Chapter 6
ALIAS (DUT)

ALIAS - Device Unit Types (DUT)

Overview

Type:	ALIAS (DUT)
Available as of:	V1.0.0.0
Inherits from:	–

Description

The ALIAS `RequestWstring` and `UserData` combines complex data structures into a data type to help simplify the user interface.

`RequestWstring`

Name	Data type	Description
`RequestWstring`	ARRAY [1..GPL.Gc_uiMaxRequest] OF WSTRING(GPL.Gc_uiRequestWstringLength)	The `RequestWstring` is an ALIAS for an array of WSTRINGS. The request STRING contains the SQL query request. Any SQL request must be divided into individual strings that do not exceed a length of 200 characters each. Adapt the size of the global parameters *(see page 41)* `Gc_uiMaxRequest` and `Gc_uiRequestWstringLength` according to the length of the SQL requests that you use in your application.

`UserData`

Name	Data type	Description
`UserData`	ARRAY [1..GPL.Gc_uiMaxRows, 1..GPL.Gc_uiMaxColumns] OF WSTRING(GPL.Gc_uiTableWstringLength)	The `UserData` is an array of WSTRING that must be available on the controller for storing the SQL data read from the database. Adapt the size of the global parameters *(see page 41)* `Gc_uiMaxRows`, `Gc_uiMaxColums`, and `GPL.Gc_uiTableWstringLength` according to the maximum of size of `UserData` to receive in your application.

Part III
Global Variables

What Is in This Part?

This part contains the following chapters:

Chapter	Chapter Name	Page
7	Global Constants List	39
8	Global Parameter List	41

Chapter 7
Global Constants List

Global Constants List (GCL)

Overview

Type:	Global constants
Available as of:	V1.0.0.0

Description

The global constants list contains the global constants of the SqlRemoteAccess library.

Global Constants

Variable	Data type	Value	Description
`Gc_sLibraryVersion`	STRING[80]	Vx.x.x.0[1]	Library version
[1] This value varies to indicate the version of the library.			

Chapter 8
Global Parameter List

Global Parameter List (GPL)

Overview

Type:	Global parameters
Available as of:	V1.0.0.0

Description

The global parameter list contains the global parameters of the SqlRemoteAccess library. They can be overwritten specifically for your project in the **Library Manager**.

The global parameters of the SqlRemoteAccess library allow you to define the size of the following ALIAS for data buffers:

- `UserData` *(see page 35)* that must be available in the controller for storing the SQL data read from the SQL database.
- `RequestData` *(see page 35)* that must be available in the controller in order to segment the SQL requests into separate strings of a maximum of 200 characters before they are sent to the SQL database.

Global Parameters

Variable	Data type	Value range	Description
`Gc_uiIpStringSize`	UINT	–	Upper limit of the WSTRING for the element `wsGwIpaddress` in the `ST_ConnectionSettings` structure *(see page 33)*. Default value: 15
`Gc_uiMaxRows`	UINT	1...50	Maximum number of rows for the ARRAY of user data *(see page 48)*. Default value: 20
`Gc_uiMaxColumns`	UINT	1...50	Maximum number of columns for the ARRAY of user data *(see page 48)*. Default value: 10
`Gc_uiTableWstringLength`	UINT	20...200	Length of the WSTRING in the ARRAY of user data *(see page 48)*. Default value: 200
`Gc_uiRequestWstringLength`	UINT	20...200	Length of the WSTRING in the ARRAY of request data. Configure a value that corresponds to the length of the SQL requests that you use in your application Default value: 200
`Gc_uiMaxRequest`	UINT	1...40	Upper limit of the ARRAY of request data. Configure a value that corresponds to the length of the SQL requests that you use in your application Default value: 20
`Gc_uiRecBufferLength`	UINT	512...2000	Length of the response buffer in bytes. Default value: 1000
`Gc_uiSendBufferLength`	UINT	512...2000	Length of the send buffer in bytes. Default value: 1000

NOTE: The variables `Gc_uiRecBufferLength` and `Gc_uiSendBufferLength` define the size of the internal buffers for sending and receiving data. The sizes of these buffers determine the number of cycles that are required to transfer a complete request or response between the controller and the SQL Gateway. Those values you chose may have an impact on the execution time of SQL services and memory, and thus on the overall performance of your application.

Part IV
Program Organization Units (POU)

What Is in This Part?

This part contains the following chapters:

Chapter	Chapter Name	Page
9	Function Blocks	45
10	Functions	51

Chapter 9
Function Blocks

What Is in This Chapter?

This chapter contains the following topics:

Topic	Page
FB_SqlDbRead	46
FB_SqlDbWrite	49

FB_SqlDbRead

Overview

Type:	Function block
Available as of:	V1.0.0.0

Task

The FB_SqlDbRead function block is used to perform SQL requests that read data from the SQL database. The return data is provided in a two-dimensional array of data whose size is defined with global parameters *(see page 42)*.

Functional Description

The FB_SqlDbRead function block is the user-interface for reading data from the SQL database.

After a rising edge on i_xExecute has been detected, a connection to the SQL Gateway is established using the parameters defined in the structure ST_ConnectionSettings. As soon as the connection has been established, the function block is capable of sending an SQL request to the SQL database.

As long as the function block is executed, the output q_xBusy is set to TRUE. After a command has been successfully completed, the output q_xDone is set to TRUE.

Status messages and diagnostic information are provided using the outputs q_xError (TRUE if an error has been detected), q_etResult, and q_etResultMsg.

Interface

Input	Data type	Description
i_xExecute	BOOL	The function block performs an SQL request in order to read data from the SQL database upon rising edge of this input.
i_refRequestWstring	REFERENCE TO [RequestWstring] *(see page 35)*	Reference to the request data that contains one SQL query request (such as Select * from DB limit 10;). Any SQL request must be divided into individual strings that do not exceed a length of 200 characters each. Adapt the size of the global parameters *(see page 42)* Gc_uiMaxRequest and Gc_uiRequestWstringLength according to the length of the SQL requests that you use in your application. **NOTE:** To concatenate WSTRINGS, use the WCONCAT function of Standard64 library.
i_uiNumOfWstrings	UINT	The number of needed WSTRINGS that contain the split SQL request. The maximum number is limited by the global parameter *(see page 42)* Gc_uiMaxRequest.
i_refUserData	REFERENCE TO [UserData] *(see page 35)*	Reference to the UserData *(see page 48)* that must be available on the controller for storing the SQL data read from the database.

In_Out	Data type	Description
iq_stConnSettings	ST_ConnectionSettings *(see page 33)*	Contains the information for connecting to an SQL Gateway and information on the SQL database.

Output	Data type	Description
q_xBusy	BOOL	If this output is set to TRUE, the function block execution is in progress.
q_xDone	BOOL	If this output is set to TRUE, the execution has been completed successfully.
q_xError	BOOL	If this output is set to TRUE, an error has been detected. For details, refer to q_etResult and q_etResultMsg.
q_etResult	ET_Result	Provides diagnostic and status information.
q_sResultMsg	STRING[255]	Provides additional diagnostic and status information.
q_uiRetNumOfRows	UINT	Number of rows in the returning data. This output is updated with the number of records which was received from the SQL database.
q_uiRetNumOfColumns	UINT	Number of columns in the returning data. This output is updated with the number of records which was received from the SQL database.

Defining an ARRAY of User Data

A 2-dimensional ARRAY must be available on the controller for intermediate storage of SQL data read from the database. The two-dimensional ARRAY is defined in ALIAS UserData *(see page 35)*.

The size of the ARRAY can be adapted via the global parameters *(see page 41)* Gc_uiMaxRows, Gc_uiMaxColumns, and Gc_uiTableWstringLength.

When you configure these parameters, consider the amount of SQL data that you expect to be received. Before data transfer is started, SQL data is segmented according to the size of this buffer.

If the SQL data that is received exceeds the size of the ARRAY, the SQL data transfer is stopped and the function block signals a detected error.

FB_SqlDbWrite

Overview

Type:	Function block
Available as of:	V1.0.0.0

Task

The FB_SqlDbWrite function block is used to perform SQL requests that update or modify the SQL database. The requests do not return any data.

Functional Description

The FB_SqlDbWrite function block is the user-interface for updating or modifying the SQL database.

After a rising edge on i_xExecute has been detected, a connection to the SQL Gateway is established using the parameters defined in the structure ST_ConnectionSettings *(see page 33)*. As soon as the connection has been established, the function block is capable to send one SQL request (given to input i_refRequestWstring) to the SQL database.

As long as the function block is executed, the output q_xBusy is set to TRUE. After a command has been successfully completed, the output q_xDone is set to TRUE.

Status messages and diagnostic information are provided using the outputs q_xError (TRUE if an error has been detected), q_etResult, and q_etResultMsg.

Interface

Input	Data type	Description
i_xExecute	BOOL	The function block performs an SQL request in order to update or modify the SQL database upon rising edge of this input.
i_refRequestWstring	REFERENCE TO [RequestWstring] *(see page 35)*	Reference to the request data that contains one SQL update request. The following SQL query types are supported: • INSERT INTO • UPDATE • DELETE FROM • CREATE TABLE • CREATE VIEW • CREATE INDEX • ALTER TABLE • DROP TABLE • TRUNCATE TABLE Any SQL request must be divided into individual strings that do not exceed a length of 200 characters each. Adapt the size of the global parameters *(see page 42)* Gc_uiMaxRequest and Gc_uiRequestWstringLength according to the length of the SQL requests that you use in your application. **NOTE:** To concatenate WSTRINGS, use the CONCAT function of Standard64 library.
i_uiNumOfWstrings	UINT	The number of needed WSTRINGS that contain the split SQL request. The maximum number is limited by the global parameter *(see page 42)* Gc_uiMaxRequest.

In_Out	Data type	Description
iq_stConnSettings	ST_ConnectionSettings *(see page 33)*	Contains the information for connecting to an SQL Gateway and information on the SQL database.

Output	Data type	Description
q_xBusy	BOOL	If this output is set to TRUE, the function block execution is in progress.
q_xDone	BOOL	If this output is set to TRUE, the execution has been completed successfully.
q_xError	BOOL	If this output is set to TRUE, an error has been detected. For details, refer to q_etResult and q_etResultMsg.
q_etResult	ET_Result	Provides diagnostic and status information.
q_sResultMsg	STRING[255]	Provides additional diagnostic and status information.

Chapter 10
Functions

FC_EtResultToString

Overview

Type:	Function
Available as of:	V1.0.0.0
Inherits from:	–
Implements:	–

Task

Convert an enumeration element of type ET_Result to a variable of type STRING.

Functional Description

Using the function FC_EtResultToString, you can convert an enumeration element of type ET_Result to a variable of type STRING.

Interface

Input	Data type	Description
i_etResult	ET_Result	Enumeration with the result.

Return Value

Data type	Description
STRING(80)	The ET_Result converted to text.

Appendices

Appendix A
Function and Function Block Representation

Overview

Each function can be represented in the following languages:

- IL: Instruction List
- ST: Structured Text
- LD: Ladder Diagram
- FBD: Function Block Diagram
- CFC: Continuous Function Chart

This chapter provides functions and function blocks representation examples and explains how to use them for IL and ST languages.

What Is in This Chapter?

This chapter contains the following topics:

Topic	Page
Differences Between a Function and a Function Block	56
How to Use a Function or a Function Block in IL Language	57
How to Use a Function or a Function Block in ST Language	61

Differences Between a Function and a Function Block

Function

A function:

- is a POU (Program Organization Unit) that returns one immediate result.
- is directly called with its name (not through an instance).
- has no persistent state from one call to the other.
- can be used as an operand in other expressions.

Examples: boolean operators (`AND`), calculations, conversion (`BYTE_TO_INT`)

Function Block

A function block:

- is a POU (Program Organization Unit) that returns one or more outputs.
- needs to be called by an instance (function block copy with dedicated name and variables).
- each instance has a persistent state (outputs and internal variables) from one call to the other from a function block or a program.

Examples: timers, counters

In the example, `Timer_ON` is an instance of the function block `TON`:

```
PROGRAM MyProgram_ST
VAR
    Timer_ON: TON;  // Function Block Instance
    Timer_RunCd: BOOL;
    Timer_PresetValue: TIME := T#5S;
    Timer_Output: BOOL;
    Timer_ElapsedTime: TIME;
END_VAR
```

```
Timer_ON(
    IN:=Timer_RunCd,
    PT:=Timer_PresetValue,
    Q=>Timer_Output,
    ET=>Timer_ElapsedTime);
```

How to Use a Function or a Function Block in IL Language

General Information

This part explains how to implement a function and a function block in IL language.

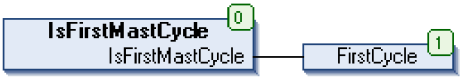
Functions `IsFirstMastCycle` and `SetRTCDrift` and Function Block `TON` are used as examples to show implementations.

Using a Function in IL Language

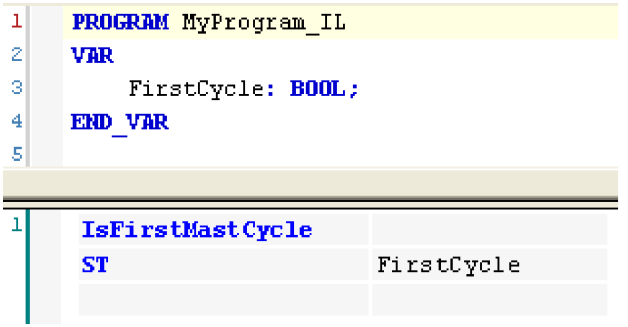
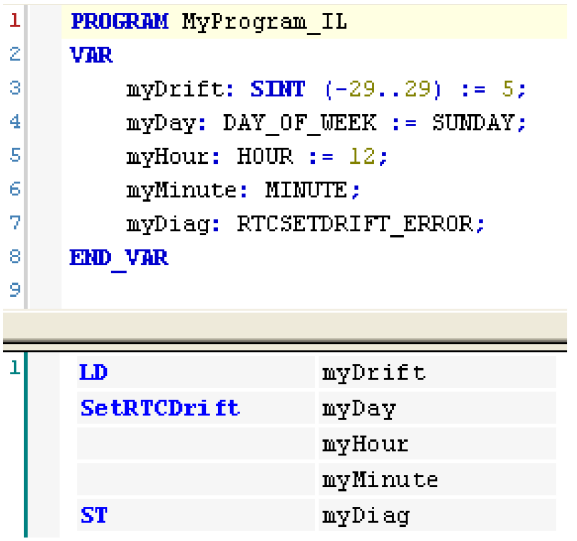
This procedure describes how to insert a function in IL language:

Step	Action
1	Open or create a new POU in Instruction List language. **NOTE:** The procedure to create a POU is not detailed here. For more information, refer to Adding and Calling POUs *(see SoMachine, Programming Guide)*.
2	Create the variables that the function requires.
3	If the function has 1 or more inputs, start loading the first input using LD instruction.
4	Insert a new line below and: ● type the name of the function in the operator column (left field), or ● use the **Input Assistant** to select the function (select **Insert Box** in the context menu).
5	If the function has more than 1 input and when Input Assistant is used, the necessary number of lines is automatically created with ??? in the fields on the right. Replace the ??? with the appropriate value or variable that corresponds to the order of inputs.
6	Insert a new line to store the result of the function into the appropriate variable: type ST instruction in the operator column (left field) and the variable name in the field on the right.

To illustrate the procedure, consider the Functions `IsFirstMastCycle` (without input parameter) and `SetRTCDrift` (with input parameters) graphically presented below:

Function	Graphical Representation
without input parameter: `IsFirstMastCycle`	
with input parameters: `SetRTCDrift`	

In IL language, the function name is used directly in the operator column:

Function	Representation in POU IL Editor
IL example of a function without input parameter: `IsFirstMastCycle`	
IL example of a function with input parameters: `SetRTCDrift`	

Using a Function Block in IL Language

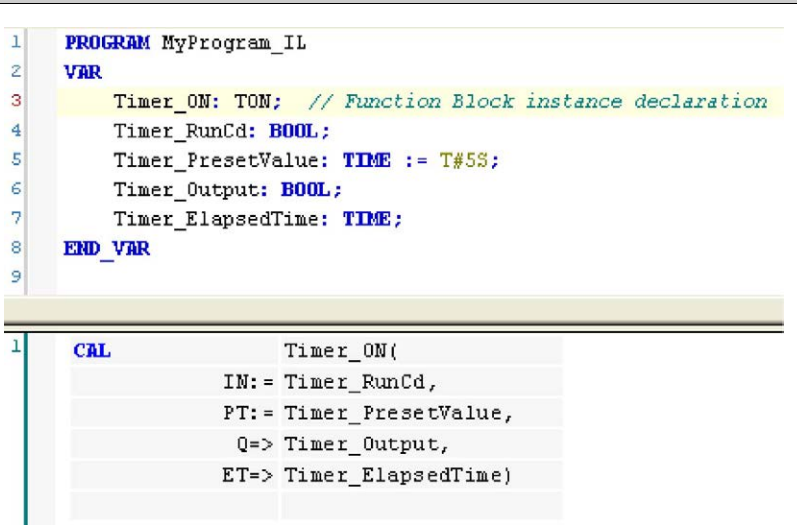
This procedure describes how to insert a function block in IL language:

Step	Action
1	Open or create a new POU in Instruction List language. **NOTE:** The procedure to create a POU is not detailed here. For more information, refer to Adding and Calling POUs *(see SoMachine, Programming Guide)*.
2	Create the variables that the function block requires, including the instance name.
3	Function Blocks are called using a CAL instruction: ● Use the **Input Assistant** to select the FB (right-click and select **Insert Box** in the context menu). ● Automatically, the CAL instruction and the necessary I/O are created. Each parameter (I/O) is an instruction: ● Values to inputs are set by ":=". ● Values to outputs are set by "=>".
4	In the CAL right-side field, replace ??? with the instance name.
5	Replace other ??? with an appropriate variable or immediate value.

To illustrate the procedure, consider this example with the TON Function Block graphically presented below:

Function Block	Graphical Representation
TON	

In IL language, the function block name is used directly in the operator column:

Function Block	Representation in POU IL Editor
TON	

```
PROGRAM MyProgram_IL
VAR
    Timer_ON: TON;  // Function Block instance declaration
    Timer_RunCd: BOOL;
    Timer_PresetValue: TIME := T#5S;
    Timer_Output: BOOL;
    Timer_ElapsedTime: TIME;
END_VAR
```

```
CAL             Timer_ON(
          IN:=  Timer_RunCd,
          PT:=  Timer_PresetValue,
           Q=>  Timer_Output,
          ET=>  Timer_ElapsedTime)
```

How to Use a Function or a Function Block in ST Language

General Information

This part explains how to implement a Function and a Function Block in ST language.

Function `SetRTCDrift` and Function Block `TON` are used as examples to show implementations.

Using a Function in ST Language

This procedure describes how to insert a function in ST language:

Step	Action
1	Open or create a new POU in Structured Text language. **NOTE:** The procedure to create a POU is not detailed here. For more information, refer to Adding and Calling POUs *(see SoMachine, Programming Guide)*.
2	Create the variables that the function requires.
3	Use the general syntax in the **POU ST Editor** for the ST language of a function. The general syntax is: `FunctionResult:= FunctionName(VarInput1, VarInput2,.. VarInputx);`

To illustrate the procedure, consider the function `SetRTCDrift` graphically presented below:

Function	Graphical Representation
`SetRTCDrift`	

The ST language of this function is the following:

Function	Representation in POU ST Editor

`SetRTCDrift`	`PROGRAM MyProgram_ST`
	`VAR myDrift: SINT(-29..29) := 5;`
	`myDay: DAY_OF_WEEK := SUNDAY;`
	`myHour: HOUR := 12;`
	`myMinute: MINUTE;`
	`myRTCAdjust: RTCDRIFT_ERROR;`
	`END_VAR`
	`myRTCAdjust:= SetRTCDrift(myDrift, myDay, myHour, myMinute);`

Using a Function Block in ST Language

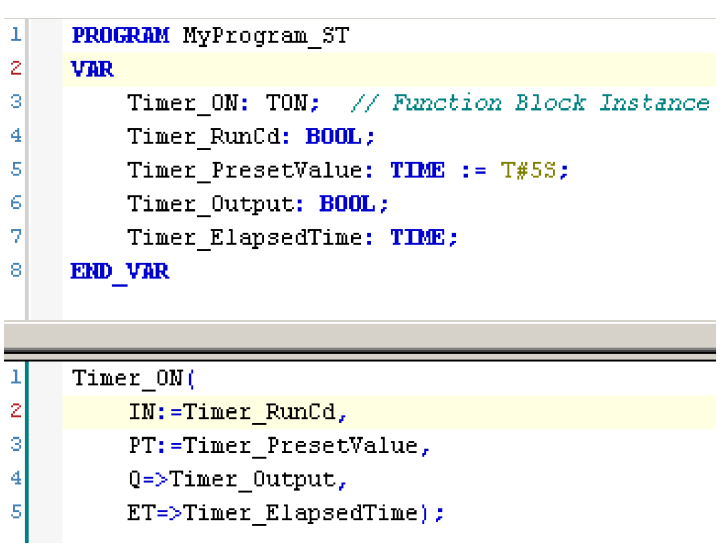
This procedure describes how to insert a function block in ST language:

Step	Action
1	Open or create a new POU in Structured Text language. **NOTE:** The procedure to create a POU is not detailed here. For more information on adding, declaring and calling POUs, refer to the related documentation *(see SoMachine, Programming Guide)*.
2	Create the input and output variables and the instance required for the function block: ● Input variables are the input parameters required by the function block ● Output variables receive the value returned by the function block
3	Use the general syntax in the **POU ST Editor** for the ST language of a Function Block. The general syntax is: `FunctionBlock_InstanceName(Input1:=VarInput1, Input2:=VarInput2,... Ouput1=>VarOutput1, Ouput2=>VarOutput2,...);`

To illustrate the procedure, consider this example with the `TON` function block graphically presented below:

Function Block	Graphical Representation
`TON`	

This table shows examples of a function block call in ST language:

Function Block	Representation in POU ST Editor
TON	

```
PROGRAM MyProgram_ST
VAR
    Timer_ON: TON;  // Function Block Instance
    Timer_RunCd: BOOL;
    Timer_PresetValue: TIME := T#5S;
    Timer_Output: BOOL;
    Timer_ElapsedTime: TIME;
END_VAR
```

```
Timer_ON(
    IN:=Timer_RunCd,
    PT:=Timer_PresetValue,
    Q=>Timer_Output,
    ET=>Timer_ElapsedTime);
```

Glossary

A

application

A program including configuration data, symbols, and documentation.

B

byte

A type that is encoded in an 8-bit format, ranging from 00 hex to FF hex.

C

CFC

(*continuous function chart*) A graphical programming language (an extension of the IEC 61131-3 standard) based on the function block diagram language that works like a flowchart. However, no networks are used and free positioning of graphic elements is possible, which allows feedback loops. For each block, the inputs are on the left and the outputs on the right. You can link the block outputs to the inputs of other blocks to create complex expressions.

configuration

The arrangement and interconnection of hardware components within a system and the hardware and software parameters that determine the operating characteristics of the system.

controller

Automates industrial processes (also known as programmable logic controller or programmable controller).

E

expansion bus

An electronic communication bus between expansion I/O modules and a controller.

F

FB

(*function block*) A convenient programming mechanism that consolidates a group of programming instructions to perform a specific and normalized action, such as speed control, interval control, or counting. A function block may comprise configuration data, a set of internal or external operating parameters and usually 1 or more data inputs and outputs.

function block diagram

One of the 5 languages for logic or control supported by the standard IEC 61131-3 for control systems. Function block diagram is a graphically oriented programming language. It works with a list of networks where each network contains a graphical structure of boxes and connection lines representing either a logical or arithmetic expression, the call of a function block, a jump, or a return instruction.

I

I/O

(*input/output*)

IL

(*instruction list*) A program written in the language that is composed of a series of text-based instructions executed sequentially by the controller. Each instruction includes a line number, an instruction code, and an operand (refer to IEC 61131-3).

INT

(*integer*) A whole number encoded in 16 bits.

L

LD

(*ladder diagram*) A graphical representation of the instructions of a controller program with symbols for contacts, coils, and blocks in a series of rungs executed sequentially by a controller (refer to IEC 61131-3).

P

POU

(*program organization unit*) A variable declaration in source code and a corresponding instruction set. POUs facilitate the modular re-use of software programs, functions, and function blocks. Once declared, POUs are available to one another.

program

The component of an application that consists of compiled source code capable of being installed in the memory of a logic controller.

S

SQL

SQL (Structured Query Language) is a programming language for managing data stored in relational database management systems.

ST

(*structured text*) A language that includes complex statements and nested instructions (such as iteration loops, conditional executions, or functions). ST is compliant with IEC 61131-3.

V

variable

A memory unit that is addressed and modified by a program.

Index

A

ALIAS - device unit types (DUT), *35*

E

ET_Result, *29*
- ClientNotConnected, *29*
- CommandExecutionIssue, *32*
- ConnectionNameNotExisting, *31*
- ConnectionNotIdle, *30*
- ConnectionTimedOut, *29*
- DataBufferTooSmall, *30*
- DbConfigurationInvalid, *31*
- DbConnectionNameNull, *29*
- DbConnectionNotAvailable, *31*
- GplMaxColumnsExceedsLimits, *31*
- GplMaxRecBufferExceedsLimits, *31*
- GplMaxRequestExceedsLimits, *31*
- GplMaxRowsExceedsLimits, *31*
- GplMaxSendBufferExceedsLimits, *31*
- GplRequestWstringExceedsLimits, *31*
- GplTableWstringExceedsLimits, *31*
- InvalidIP, *29*
- InvalidPort, *30*
- InvalidRequestWstring, *31*
- InvalidUserDataWstring, *31*
- InvalidWstringNumber, *31*
- NoDataReturned, *30*
- NoQueryResultAvailable, *32*
- NotConnected, *30*
- Ok, *29*
- SocketNotClosed, *30*
- SqlDatabaseBusy, *30*
- SqlGwInvalidLicense, *32*
- SqlGwProtocolError, *32*
- SqlGwToPlcCorruptedMessage, *30*
- SqlGwUnexpectedError, *32*
- TcpClientIssue, *29*
- TimeoutExpired, *30*
- TransactionNumMismatch, *30*
- TransmissionNumMismatch, *30*

F

FB_SqlDbRead, *46*
FB_SqlDbWrite, *49*
FC_EtResultToString, *51*
functions
- differences between a function and a function block, *56*
- how to use a function or a function block in IL language, *57*
- how to use a function or a function block in ST language, *61*

G

GCL (Global Constants List)
- SqlRemoteAccess, *39*

GPL (Global Parameter List)
- SqlRemoteAccess, *41*

L

libraries
- SqlRemoteAccess, *19*

Q

qualification of personnel, *14*

S

SQL communications
- considerations, *20*

SqlRemoteAccess, *19*
- FB_SqlDbRead, *46*
- FB_SqlDbWrite, *49*
- GCL (Global Constants List), *39*
- GPL (Global Parameter List), *41*

ST_ConnectionSettings, *33*